U.S.N.A. — Trident Scholar project report; no. 302 (2003)

MULTIRESOLUTIONAL OPTIC FLOW

by

Midshipman Matthew J. Ahlert, Class of 2003
United States Naval Academy
Annapolis, Maryland

(signature)

Certification of Advisers' Approval

Associate Professor John F. Pierce
Department of Mathematics

(signature)

(date)

Professor Reza Malek-Madani
Department of Mathematics

(signature)

(date)

Acceptance for the Trident Scholar Committee

Professor Joyce E. Shade
Deputy Director of Research & Scholarship

(signature)

(date)

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	**2. REPORT DATE** 2 May 2003	**3. REPORT TYPE AND DATE COVERED**

4. TITLE AND SUBTITLE Multiresolutional optic flow	**5. FUNDING NUMBERS**
6. AUTHOR(S) Ahlert, Matthew J. (Matthew Joseph), |d1981-	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) US Naval Academy Annapolis, MD 21402	**10. SPONSORING/MONITORING AGENCY REPORT NUMBER** Trident Scholar project report no. 302 (2003)

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT This document has been approved for public release; its distribution is UNLIMITED.	**12b. DISTRIBUTION CODE**

13. ABSTRACT: The irony embedded in optic flow computation is that despite its not having been studied mathematically until about 20 years ago, humans have been performing the related calculations innately since their creation. The process can be thought of as filling in the gaps that occur between the images we see changing in time. For the human mind, the transition is automatic from what an image looks like at one instant to what it looks like at the next. Our brain intuitively generates a continuous flow of what it perceives is being viewed by the eyes. Manifested in a mathematical sense, optic flow serves as a mechanism to describe the movement of objects in a digital image sequence using a flow field of grayscale intensity. This particular study has focused upon developing a usable mathematical implementation for two of the various algorithms, which have been proposed to compute optic flow. One method relies upon calculus of variations to regularize the problem while the other utilizes a wavelet based multi-scale approach. Possible applications for such research range from data compression of video sequences to the development of a more efficient way to analyze the time varying one-dimensional acoustic imagery supplied by sonar systems.

14. SUBJECT TERMS: optic flow; digital images; acoustic imagery	**15. NUMBER OF PAGES** 62
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**

1 Abstract

The irony embedded in optic ‡ow computation is that despite its not having been studied mathematically until about 20 years ago, humans have been performing the related calculations innately since their creation. The process can be thought of as …lling in the gaps that occur between the images we see changing in time. For the human mind, the transition is automatic from what an image looks like at one instant to what it looks like at the next. Our brain intuitively generates a continuous ‡ow of what it perceives is being viewed by the eyes.

Manifested in a mathematical sense, optic ‡ow serves as a mechanism to describe the movement of objects in a digital image sequence using a ‡ow …eld of grayscale intensity. This particular study has focused upon developing a usable mathematical implementation for two of the various algorithms which have been proposed to compute optic ‡ow. One method relies upon calculus of variations to regularize the problem while the other utilizes a wavelet based multi-scale approach.

Possible applications for such research range from data compression of video sequences to the development of a more e¢cient way to analyze the time varying one-dimensional acoustic imagery supplied by sonar systems.

2 Acknowledgements

The author most likely could go on for the length of an entire thesis thanking those who have aided his endeavors throughout the course of this project. Unquestionably, these words of gratitude must begin with his advisers. To Dr. John Pierce the author owes both his sanity and his source of inspiration. Much more than an instructor, he has been both a mentor and a guide through every up and most thankfully through every down along the way. To Dr. Reza Malek-Madani, for showing that in a determined quest for knowledge and understanding, the world can retain the very same sense of promise it held during childhood. Ever it was he whose questions somehow kept the author's feet on the ground while letting his head stay in the clouds. As a collaborator on the project, thanks go to LCDR Joseph Skufca, USN, for his creativity in actually finding a project to work on, and for his efforts in helping the author tame the demons of sonar. Appreciation must also go out to Dr. Peter Turner for providing the initial confidence to undertake such a venture. To the Trident Scholar Committee and Dr. Joyce Shade for providing a tangible sense of focus when the lines began to get blurry. Thanks also to Christine Jamison and Karen Lambert in the MSC for their time, expertise, and unending patience in helping to put together an extraordinary poster in just a few days time.

A big bright smile goes to Dr. Sonia Garcia for never allowing the author not to have one upon his face. To Christopher McFadden for illustrating what wavelets really are. To Pritha Mahadevan, the only one who ever really understood. To Edward Estlin Cummings for the light in his quote. And lastly, the author's love and thanks go with all his heart to his family for being who they are. Had it not been for their undying support, this enterprise would have come to an end long ago.

3 Report Information

3.1 Table of Contents

3.2 Keywords

Optic Flow

Wavelets

Multiresolution Analysis

Multiscale Methods

Calculus of Variations

Galerkin Method

Image Sequences

Signal Processing

3.3 List of Figures

3.4 Notation

$\mathbf{x}$	$= (x_1, x_2)$ Cartesian coordinate of a pixel in a 2-D image
$I(\mathbf{x}; t)$	Grayscale intensity of the image at position $\mathbf{x}$ and time t
$\frac{\partial}{\partial x_i}$	Partial derivative with respect to x_i where $i = 1, 2$
$\frac{\partial}{\partial t}$	Partial derivative with respect to t
$\mathbf{v}(\mathbf{x}; t)$	$= (v_1, v_2) = \left(\frac{\delta x_1}{\delta t}, \frac{\delta x_2}{\delta t} \right)$ Optic Flow vector
∇I	$= \left(\frac{\partial I}{\partial x_1}, \frac{\partial I}{\partial x_2} \right)$ Gradient of I
$\overline{g(x)}$	Complex Conjugate of the function g
$\langle f, g \rangle$	$= \iint f(\mathbf{x}) \overline{g(\mathbf{x})} dx_1 dx_2$ Inner Product of f and g
ψ	Generic symbol for a 1-D wavelet
ϕ	Generic symbol for a 1-D scaling function
θ^n	2-D mother wavelet with $n = 1, .., N$ where $N = 3$ or 4
θ^1	$= \psi(x_1)\phi(x_2)$ High-Low component
θ^2	$= \phi(x_1)\psi(x_2)$ Low-High component
θ^3	$= \psi(x_1)\psi(x_2)$ High-High Component
θ^4	$= \phi(x_1)\phi(x_2)$ Low-Low Component
$\theta^n_{\mathbf{u}}$	$= \theta^n(\mathbf{x} - \mathbf{u})$ where $\mathbf{u}$ is a 2-D continuous translation index
$\theta^n_{\mathbf{u}s}(\mathbf{x})$	$= s^{-1}\theta^n \left(\frac{\mathbf{x} - \mathbf{u}}{s} \right)$ where s is a continuous scaling index. This forms a 2-D continuous mother wavelet family
$\theta^n_{j\mathbf{k}}(\mathbf{x})$	$= 2^{j/2}\theta^n \left(2^j \mathbf{x} - \mathbf{k} \right)$ where $j \in \mathbb{Z}$ and $\mathbf{k} \in \mathbb{Z}^2$. This forms a 2-D discrete mother wavelet family where $\psi^n_{j\mathbf{k}}$ is located around $\left(2^{-j}k_1, 2^{-j}k_2 \right)$ and spreads over a domain of size proportional to 2^{-j}
$L_2(\mathbb{R})$	Set of all square-integrable functions in $\mathbb{R}$

4 Overview

4.1 Introduction

What is it about motion that human beings …nd so interesting? Why are we so much more likely to follow a moving object with our eyes than we are to let them rest on something which is static? Perhaps it is because of the questions we learn to automatically ask ourselves: Where is the object going? Where is it coming from? How fast is it moving? Do I need to move as well in order to avoid its path? Answers to this cycle of questions must continually be found as one carries out the process of evading collision. Perhaps the intriguing factor about motion is that it always takes us to someplace new, someplace di¤erent from where we just were. The old Italian proverb speaks truly, "A rolling stone gathers no moss." As humans we yearn to see what we have not yet seen, and only through movement may we accomplish this goal and turn our view to the unexplored.

This same fascination with motion carries over to the use of a computer in our daily lives. Namely, it is while watching video that our minds are most captivated by the screen. Neither text, nor graphics, nor sound can convey so much information to the brain so quickly. Video is essentially the playing of image sequences and it is only here that one may watch objects moving through space in time. But how does the computer view this wealth of data? Does it assign any signi…cance to what is happening in the video?

The answer, put quite simply, is no. A computer devotes absolutely no e¤ort to understanding how the objects are moving in a digital image sequence. It merely displays frame after frame and leaves it up to the user to mentally …ll in the gaps between images in order to perceive smooth motion. For instance, let's say a video clip shows a baseball being thrown from one person to another. Even though the ball is at a certain position in one image frame and at another position in the next, the computer has no way of knowing where the ball probably was for the time in between the two images. The human mind, on the other hand, is intelligent enough to …gure out that the ball must have been somewhere between those two positions during that time. Thus our brain creates a continuous ‡ow of motion instead of seeing just a series of discrete still images being played in succession.

An interesting problem here results. Can we solve for this ‡ow in a digital sense? What kind of information would this provide? How can such a question be tied to applied mathematics? The answers lie in the concept of optic ‡ow. At the most basic level, it can be understood as a mechanism for describing the motion captured within an image sequence.

Theoretical development for this scheme has its foundation in the seminal paper by Horn and Schunck [14] and has later been expounded upon by the publications of Christophe Bernard [3], [4], [5], [6], [7]. In both cases, the authors do not provide the code they use to generate their results. Rather, they leave their theories as virtual "black boxes" which may be employed to extract information relating to the motion of objects appearing in the image sequence.

As often will happen in the undertaking of a project such as this, more questions have surfaced than were there at the start. Many riddles still remain surrounding optic ‡ow. For instance, what is the best way to solve for it? Just how sound are the physics which constitute the theory behind optic ‡ow? Finally, how useful is the information gained when applied to a practical context? Despite such issues still not being resolved, new ground has certainly been broken in the quest to present optic ‡ow as a …eld of applied mathematics which is open to experimentation. A concrete path for implementing the idea has …nally been laid down.

4.2 Outline of Approach

The body of this paper is divided into …ve main sections. The …rst serves to illustrate the details of optic ‡ow, how the concept is applied to an image sequence, what kind of equation is being solved, and exactly what information is being extracted. As our study has taken two completely di¤erent directions in trying to solve for this optic ‡ow, the next two sections of the paper are devoted to an explanation of each method. At the outset of the project, we sought to treat the problem using a wavelet based multiscale approach in the same fashion as Bernard [3], [4], [5], [6], [7]. The actual implementation required to do this proved itself much more complicated than originally envisioned. Giving up on the problem was not an option, however, and so the decision was made to begin a new approach in parallel to our ongoing e¤orts. This attempt relied upon the technique of calculus of variations in a similar manner as Horn and Schunck [14]. As our research currently stands, the protocol for implementing each method has been developed to the extent that they can operate on simpli…ed approximations to motion being captured in an image sequence. The fourth section of the paper evaluates the results attained using these two approaches. Conclusions are presented in the …fth and …nal section of the paper. Various issues will be discussed here including: which method appears to work better; what opportunities exist to adjust each model to better suit the problem; what issues are left unresolved; what are some of the possible future applications for optic ‡ow; and how does the overall idea behind optic ‡ow stand against other problems examined in applied mathematics.

4.3 Desired Result

The theory behind this project is both powerful and highly desirable for application to a context where determination of an object's motion is the foremost goal. Without a tangible way of utilizing it, however, a theory doesn't amount to having much value at all. This is especially true for the Navy and the military as a whole which places an indisputable requirement on practicality. Therefore, the primary focus of the research conducted for this project has been to produce a deliverable which harnesses the potential of this theory and sets it down in a usable code.

As a second bene…t accompanying our study, we have performed a mathematical comparative analysis on the di¤erent methods for implementation. In each case, observations were made regarding the advantages and shortcomings of that particular approach . E¤ectively, this process may be considered the optimization of the tools we have created for practical use. This is the very crux of the original project proposed because it answers the questions of how one might take an existing mathematical theory and apply it. The knowledge gained throughout the development of such implementation has been an invaluable reward to the writer and hopefully it will serve as the same for those who seek to build upon its foundations.

5 Optic Flow

5.1 Concept Introduced

The irony embedded in optic ‡ow computation is that despite its not having been studied mathematically until close to 20 years ago, human beings have been performing the related calculations innately since their creation. In its most common sense, optic ‡ow measurement is the description of images as they change in space and time. The process can be thought of as …lling in the gaps that occur between the instantaneous images we see. For the human mind, the transition is automatic from what an image looks like at one instant to what it looks like at the next. Our brain intuitively generates a continuous ‡ow of what it perceives is being viewed by the eyes [4].

The digital analogy is not so simple. Image sequences, or video, is treated by a computer as a set of discrete pictures which when cycled through quickly, create for the viewer the e¤ect of a scene changing as time progresses. If the images were not recorded at a rapid enough rate, when played back, the video will appear blurred and choppy. This phenomenon occurs because the computer does not have the inherent intelligence to say that what was at one place at one time must have gotten to its next position along a relatively smooth and logical path. It cannot automatically assume that for those instants when it does not have a displayable image, the objects in motion were at some point in between the recorded positions. This is exactly the problem which spawned the concept of optic ‡ow [4].

In one of the most de…nitive works analyzing the subject, Christophe Bernard describes optic ‡ow as “a visual displacement ‡ow …eld that can be used to explain changes in an image sequence” [3], [4]. In other words it can be thought of as a vector …eld with its arrows pointing in the direction of movement for the objects in motion within the video. He provides a very clear illustration of this in Figures 1 and 2:

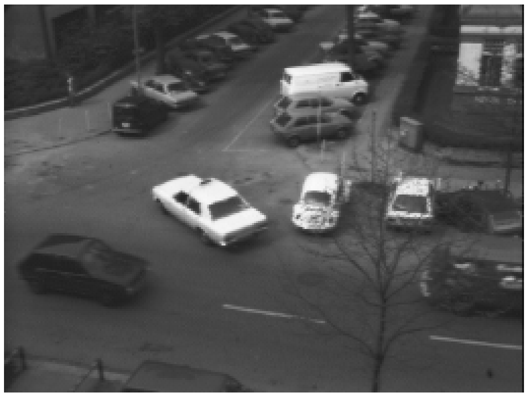

Figure 1: Car Scene

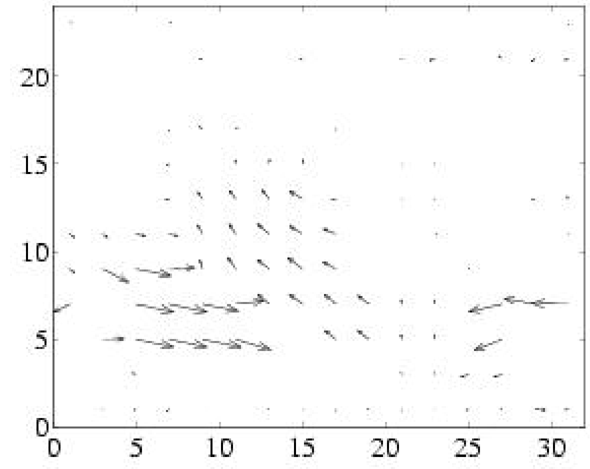

Figure 2: Measured Optic Flow

Here three cars are in motion: a light colored taxi making a turn onto a side street, as well as a dark car and a dark van which are driving in opposing directions on the main street. The corresponding measured image ‡ow graph shows a high concentration of vector arrows in the location of each of the three moving objects representing their direction of travel. In those areas of the image where there is no motion, e.g. the parked cars and building, very few arrows appear and none have signi…cant magnitude. Our goal is to be able to solve for these vectors based on a mathematical model for what motion is going on in the image sequence.

Observing this vector …eld, we see that the velocity of the grayscale intensity is being represented from an Eulerian perspective since it gives us a snapshot of the optic ‡ow as a function of pixel position. This allows us to visualize how the image sequence is changing at each instant in time as opposed to a Lagrangian approach which would display a vector …eld that would trace the path of individual objects as they moved throughout time [17, pg. 304].

The immediate question that must be asked is exactly how a computer is supposed to correctly detect the movement of objects in an image sequence. Surprisingly, the answer is not overly complicated. Every image stored digitally can be thought of as a three-dimensional surface with the x and y coordinates describing the location of a pixel within the image and the z coordinate relating to the grayscale, or intensity, of that speci…c pixel. This scheme is easily visualized by examining Figures 3 and 4:

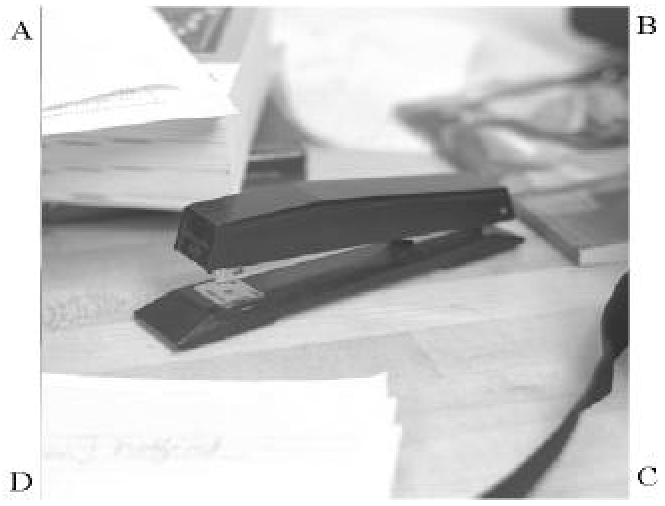

Figure 3: Sample Grayscale Image

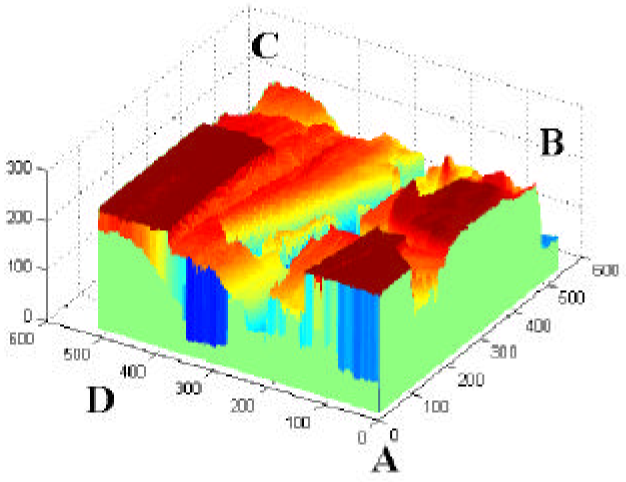

Figure 4: Digital Representation of Grayscale Image

Within the picture of the stapler sitting on the desk, there exist varying grayscale intensities. These directly correspond to the di¤erent heights of the surface portrayed in Figure 4. Where the image in Figure 3 is white, the surface in Figure 4 is very high. Conversely, where the image is black, the surface is low. NOTE: The color appearing in Figure 4 serves merely to visually accentuate the shape of the surface and has no correlation to color appearing or not appearing in the image. Also, the labels A, B, C, and D appear only as a guide for the reader to …nd the corresponding corners between the image and the surface.

As time progresses and objects move within a scene, the intensities of the individual pixels will invariably change. This movement is visualized by the computer as the three-dimensional surface changing shape. It is here that mathematics enters the problem as surfaces and their changing shape in space and time are describable using partial di¤erential equations (PDEs).

5.2 Process Illustrated

The task now becomes to …t a mathematical model to the concept of optic ‡ow. Our …rst step in this process is to decide on a set of assumptions which make the problem more approachable at …rst glance. Two critical assumptions made by Horn & Schunck and Bernard are: the objects appearing in the image are receiving uniform illumination (i.e. the ambient lighting is not changing); and the grayscale intensity of a particular point on the object is constant over local space and time [14], [3], [4], [5], [6], [7]. Although this may seem like a gross oversimpli…cation of real-life conditions, these constraining assumptions actually make sense when taken in terms of the practicality of optic ‡ow.

At its root, optic ‡ow is a mechanism for representing the motion of objects in an image sequence using vector arrows. In order to …nd these arrows, we choose a starting or "base" image in the sequence in which all the objects pictured are said to be in their original position. As time progresses, the objects move and so their pixel positions change from image frame to image frame. The concept of optic ‡ow is to track these moving objects and assign to this motion a set of vector arrows from frame to frame. As is the case with any mathematical model, there will be some error in optic ‡ow's ability to predict. There will be a di¤erence between the actual image frames and what is predicted from the base image and the ‡ow …elds. This di¤erence is known as the correction factor. Since every frame is known, the correction factor can be solved for exactly and so perfect reconstruction can occur for subsequent images using the base image, the optic ‡ow …eld, and the correction factor. In essence, this says that we can project into the future of the image sequence from a certain starting point in time.

Now, we ask what happens when one of our constraining assumptions is violated? This could very realistically happen when the illumination in the image changes from frame to frame and one of the objects in motion now appears di¤erently in its shading/grayscale intensity. We may no longer use the original base image to project forward in time with optic ‡ow. The overall process, however, doesn't come to a permanent halt. We merely begin over again by setting the next frame in the sequence to be the new base image.

From there we begin tracking the motion of objects from their new "original" position. Figure 5 illustrates this process.

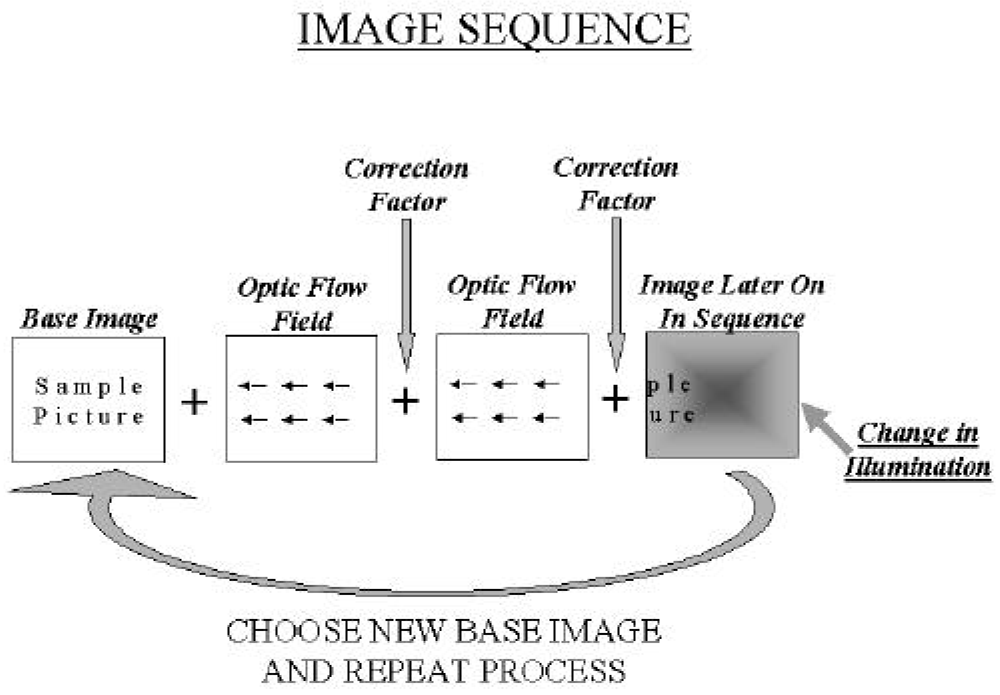

Figure 5: Schematic for Optic Flow Process

There are two ways of representing the motion of objects smoothly in an image sequence, either use more image frames per unit time or actually track the object movement using vector arrows in optic ‡ow. According to the method just described, optic ‡ow provides the distinct advantage of using much less digital storage space. Byte requirements for vector …elds/correction factors as opposed to full images are markedly less. Thus consider an average video sequence which uses 32 frames per second. Even if it is undergoing rapid illumination changes and a new base image is having to be chosen about once a second, optic ‡ow is still able to replace about 31 images with vector …elds [5], [6].

5.3 Governing Equation

Having stepped through that process, let us now turn to the development of the governing partial di¤erential equation (PDE) for optic ‡ow [14]. Starting from the constraining assumption that the grayscale intensity of an object is the same over local space and time (and using the notation previously de…ned), we arrive at the following equivalence relation:

$$I(x_1, x_2, t) = I(x_1 + \delta x_1, x_2 + \delta x_2, t + \delta t). \qquad (1)$$

In expanding the right hand side of Equation (1) around the point (x_1, x_2, t) we see that

$$I(x_1, x_2, t) = I(x_1, x_2, t) + \delta x_1 \frac{\partial I}{\partial x_1} + \delta x_2 \frac{\partial I}{\partial x_2} + \delta t \frac{\partial I}{\partial t} + \varepsilon. \qquad (2)$$

where ε is the error term. If we consider the limit of going over the most localized space and time, then ε ! 0 . The next step is to subtract $I(x_1, x_2, t)$ from both sides of Equation (2) and then divide both sides by δt . This yields

$$0 = \tfrac{\delta x_1}{\delta t} \tfrac{\partial I}{\partial x_1} + \tfrac{\delta x_2}{\delta t} \tfrac{\partial I}{\partial x_2} + \tfrac{\partial I}{\partial t}.$$

which is equivalent in the prede…ned notation to be

$$0 = \frac{\partial I}{\partial x_1} v_1 + \frac{\partial I}{\partial x_2} v_2 + \frac{\partial I}{\partial t}. \qquad (3)$$

As Equation (3) is the governing PDE for optic ‡ow, this represents the beginning of our analysis.

The image intensity function I is known at each pixel x at each time t. Therefore we may state that the partial derivatives of the intensity $\frac{\partial I}{\partial x_1}$, $\frac{\partial I}{\partial x_2}$, and $\frac{\partial I}{\partial t}$ can be found. Thus in this governing equation for optic ‡ow, also known as the intensity constraint equation, we see that there are two unknowns, namely the components of the velocity v_1 and v_2. In a single linear equation such as this where there are two unknowns, the problem is said to be ill-posed. We cannot solve for both components of the velocity without an additional constraint. Figure 6 illustrates this concept.

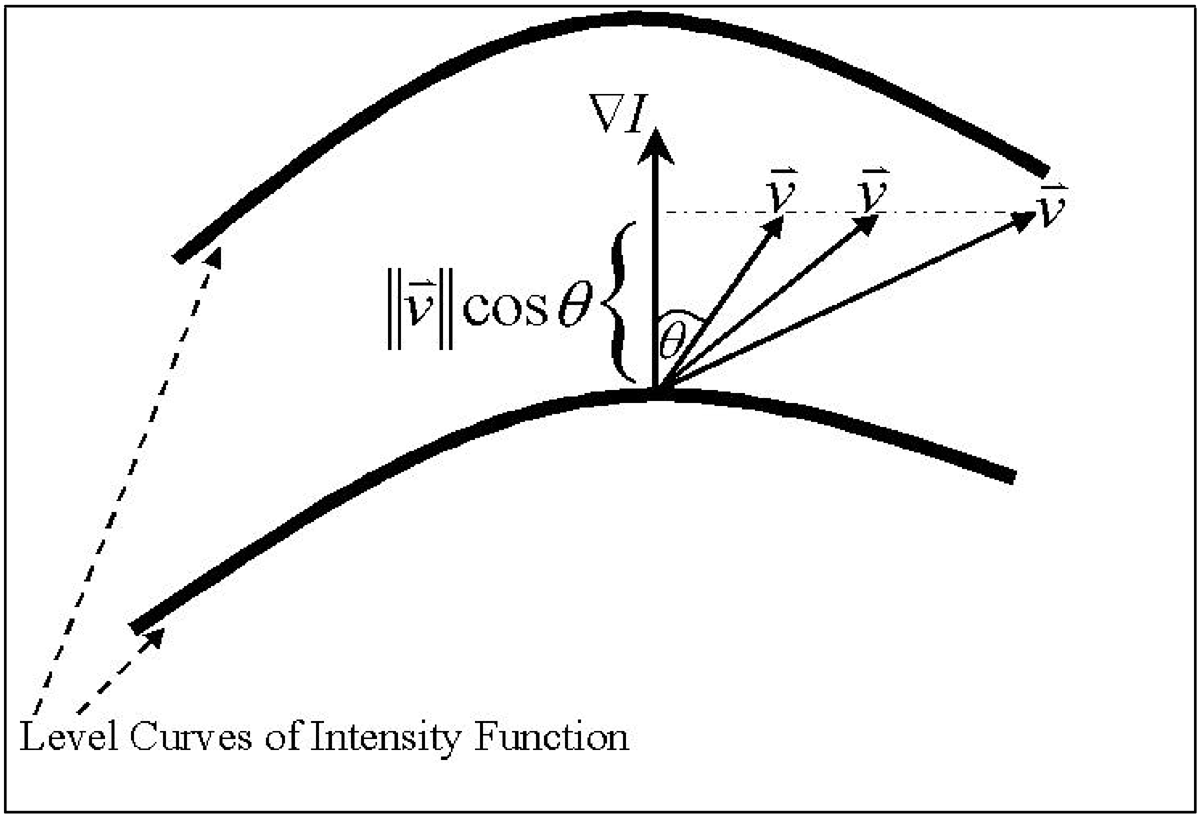

Figure 6: Ill-Posedness of Optic Flow Problem

In this example figure, a level curve of the intensity function can be thought of as a part of the image which has the same grayscale intensity throughout. By definition of gradient, the vector $\nabla I = \left(\frac{\partial I}{\partial x_1}, \frac{\partial I}{\partial x_2}\right)$ is perpendicular to the level curves. The reader will see that using the governing equation, one can always solve for the component of the optic flow which is in the direction of the gradient of the intensity function for any given point x.

Equation (3) can be rewritten,

$$\left(\frac{\partial I}{\partial x_1}, \frac{\partial I}{\partial x_2}\right) \cdot (v_1, v_2) = \nabla I \cdot \mathbf{v} = -\frac{\partial I}{\partial t}. \tag{4}$$

If we apply the definition of dot product that $\mathbf{a} \cdot \mathbf{b} = \|a\|\|b\|\cos\theta$ where θ is the angle between the two vectors, we arrive at Equation (5),

$$\nabla I \cdot \mathbf{v} = \|\nabla I\|\|\mathbf{v}\|\cos\theta = -\frac{\partial I}{\partial t}. \tag{5}$$

Dividing through by $\|\nabla I\|$ gives us the component of the optic ‡ow vector in the direction of the velocity gradient,

$$\|v\|\cos\theta = -\frac{\frac{\partial I}{\partial t}}{\|\nabla I\|}. \tag{6}$$

This component of the optic ‡ow vector, which is labeled in Figure 6, can always be solved for using the governing PDE. What must be noted from Figure 6, however, is the fact that various optic ‡ow vectors could all have this same component. The second component of the optic ‡ow (i.e. the one that is perpendicular to ∇I) cannot be solved for using only the intensity constraint equation. At least one more equation is needed to be able to …nd the complete optic ‡ow v as a unique solution. This condition is what makes the problem ill-posed as it now stands.

There are four principal approaches to making this problem one that can be considered well-posed mathematically: regularization, correlation/matching methods, spatiotemporal …ltering methods, and …l-tered di¤erential methods [3], [2]. Sections 6 and 7 of this paper examine the attempts made by Bernard and Horn & Schunck, respectively. Bernard's approach can be characterized as a projected di¤erential method which employs a multiscale technique. According to this method, he "hits" the governing equation with a wavelet basis, makes an assumption about the optic ‡ow on a local scale, and so transforms the problem into a system of matrix equations. Thus he provides a satisfactory number of equivalence relations to uniquely solve for both components of the velocity …eld [3], [4], [5], [6], [7]. In a completely di¤erent approach, Horn and Schunck rely on the addition of a smoothness constraint and the technique of calculus of variations to regularize their problem and so make it well-posed [14].

5.4 Additional Issues to Consider

Despite the existence of several valid methods for making optic ‡ow a stable and solvable problem, there remain a few additional sources of error which require addressing [27, pg. 70]. Here they will only be mentioned in brief.

The …rst source can be thought of in terms of stochastic error. Speci…cally, the digital mapping of real-life scenes to grayscale image sequences will invariably include the presence of sensor noise. There exists a disparity between the actual velocity of the objects being …lmed and what the corresponding velocity will be in the optic ‡ow …eld. This di¤erence may be described as a random noise variable which is statistical in nature. Various techniques exist to minimize the error stemming from this source [23].

Another key issue which surfaces is the fact that the governing equation for optic ‡ow neglects any interaction by second order terms in the displacement of grayscale intensity. In other words, it only accounts for ‡ow velocity and pays no attention to the e¤ects of acceleration within the ‡ow. This leads to a systematic error whenever the magnitude of the velocity is large over a localized area [27, pg. 70].

Finally, we note that error could arise from a ‡aw in the model for optic ‡ow itself. Perhaps the physics and the theory behind the governing equation don't hold and the assumption that the brightness of an object doesn't change over time (i.e. that grayscale intensity is conserved) is not valid. These problems could very well manifest themselves in a negative sense as one goes about solving for optic ‡ow depending upon the particular image sequence [27, pg. 70].

Within the scope of this paper, methods for overcoming the …rst and last of these sources of error will not be addressed in detail. However, the second issue concerning the systematic error due to large displacements is dealt with under Bernard's approach in the next section.

6 Wavelet Based Multiscale Approach

6.1 What Makes Wavelets Unique

As was stated in the previous section, Bernard attempts to overcome the ill-posedness of the problem by creating a system of matrix equations through the use of a wavelet basis. But what makes wavelets so special that they provide a better way to solve for the optic ‡ow vectors? What is special about the information wavelets can extract?

Bernard describes the utility of wavelets in the context of optic ‡ow in terms of their ability to break through the challenge posed by the aperture vs. time aliasing problem. When trying to capture the motion of an object in an image sequence, the di¢culty lies in …nding a mathematical "window" which is big enough to measure the displacement of that object between frames, and yet not so large that the motion becomes so insigni…cant that it is undetectable [3].

An illustrative example of this concept presents itself if one thinks of watching a game of baseball under two di¤erent circumstances. The …rst involves looking at the …eld through a pair of binoculars. Although one is able to zoom in and watch what sign the catcher is giving to the pitcher, it is impossible to simultaneously see that the runner on …rst base is moving to steal second. This inability to see large scale motion while looking through a small, high resolution window is referred to as the aperture problem. In e¤ect, the aperture one is looking through is not of su¢cient size to see the big picture. The second case entails watching the game using only one's normal vision. Now the viewer can see that the runner is attempting to steal the base, however, they cannot at the same time see what motion the catcher makes with his …ngers as a sign to the pitcher. Here the window is so large that small scale motion becomes insigni…cant and so is lost to the sensors of visual perception. This is known as time aliasing since small displacements become aliased out of the picture by the comparatively large and low resolution viewing window.

In terms of solving for optic ‡ow, the greatest impediment comes when large scale motion is occurring in one region of the image sequence while small scale motion is simultaneously occurring in another. The following discussion will demonstrate the properties of wavelets that allow them to overcome this problem and so avoid the limitations of time aliasing vs. aperture. Upon completion of this general overview of wavelets, we will see how Bernard applies them to the specific challenges posed by optic flow.

6.2 Wavelets and Multiresolution Analysis

As the inherent value of wavelets stems from their ability to perform a multiresolution analysis (MRA), we shall start there by defining exactly what this concept means. The basic principle behind a multiresolution analysis is to decompose a function space into a sequence of subspaces, denoted by V_j. It must be noted that the set of functions mentioned here are all those which are square integrable, namely all functions f such that $\int_{-\infty}^{\infty} (f(t))^2\, dt < \infty$. This function space is denoted by $L_2(\mathbb{R})$. When referring to the decomposition of f, it may be helpful to think of the function as a signal which is being broken down into its components occurring at different scales. The subspace V_j is defined according to its integer valued resolution j such that all details of the signal which are of scale less than 2^{-j} are suppressed [19], [28], [26], [16], [20].

A number of requirements are placed upon the subspaces V_j in order for them to form a true multiresolution analysis. The first states that V_j be contained in every subspace which is higher in resolution than j. If we define the j^{th} level approximation to $f(t)$ as $f_j(t)$ then $f_j(t) \in V_j$, which in plain language means that $f_j(t)$ belongs to the subspace V_j. Logically, we see that any details which can be captured at resolution level j must also be included with the information at a higher resolution. Therefore, V_j has to be contained in V_{j+1}, which is represented mathematically by $V_j \subset V_{j+1}$ for all integer values of j. Extending this principle, we arrive at the following nesting of subspaces: $\ldots \subset V_{j-2} \subset V_{j-1} \subset V_j \subset V_{j+1} \subset V_{j+2} \subset \ldots$ [22], [28], [19], [16].

The details which are lost in going from the approximation of f at level $j+1$ to level j are those which occur at scale $2^{-(j+1)}$. We define them by $d_j(t) = f_{j+1}(t) - f_j(t)$. From this relationship we arrive at the equivalence relation

$$f_{j+1}(t) = f_j(t) + d_j(t).$$

The corresponding subspaces can be represented accordingly,

$$V_{j+1} = V_j \oplus W_j.$$

Here the notation W_j refers to the detail space at resolution level j and W_j is orthogonal to V_j. This means that if we took the inner product between any element g in W_j and any element h in V_j, the result would equal 0:

$$\langle h, g\rangle = \int_{-\infty}^{\infty} h(t)\overline{g(t)}dt = 0.$$

This breakdown of subspaces can continue on so that we obtain Equation (7),

$$V_{j+1} = W_j \oplus V_j = W_j \oplus W_{j-1} \oplus V_{j-1} = \ldots = W_j \oplus W_{j-1} \oplus W_{j-2} \oplus \ldots \oplus W_{j-J} \oplus V_{j-J}. \tag{7}$$

If W_j is orthogonal to V_j it is also orthogonal to any subspace of V_j, including all detail spaces W_k such that $k < j$. From Equation (7), we see that our approximation space at level j, V_j, can be written as a sum of subspaces. This is the first requirement for an MRA [28], [19].

The second necessary condition is that all square integrable functions can be represented at the finest level approximation space and also that only the zero function is contained at the coarsest level of resolution. Explanation for the second of these points stems from the fact that as the resolution becomes coarser and coarser, more and more details are lost and at the limit $j \to -\infty$, only a constant function remains. In order to meet the requirement of square integrability, that function must be exactly equal to zero. Mathematically this can be represented either by the limit: $\lim_{j \to -\infty} V_j = \{0\}$ or by the infinite intersection of subspaces: $\cap_j V_j = \{0\}$. Conversely, as resolution increases, more and more details are included. Therefore, as the resolution level approaches positive infinity, the entire space of square integrable functions should be recovered. This can be written mathematically either by the limit: $\lim_{j \to \infty} V_j = L_2(\mathbb{R})$ or by the closure of the union of all subspaces: $\overline{\cup_j V_j} = L_2(\mathbb{R})$ [28], [22].

The third requirement states that all the spaces $\{V_j\}$ are scaled versions of the space V_0 which is named the central space. If we say that the function $f(t) \in V_j$, in other words there are no details in $f(t)$ which appear at scales smaller than 2^{-j}, then $f(2t)$ is the function attained by compressing $f(t)$ by a factor of 2 so that it contains no details at scales less than $2^{-(j+1)}$. Therefore, $f(2t) \in V_{j+1}$. This condition is known as scale or dilation invariance [28].

Condition number four mandates that the space V_j be translation or shift invariant. Therefore, if $f(t) \in V_0$ then also $f(t - k) \in V_0$ for all integers k. The combination of this requirement with the third one relating to scale invariance give way to the fact that if $f(t) \in V_0$, then $f(2^j t - k) \in V_j$ [19], [28], [16].

The final requirement for a multiresolution analysis is that there must exist a function ϕ such that all of its translates form an orthonormal basis for V_0. In this case, orthonormality requires two separate conditions be met, one of orthogonality and the other of normalization to 1. The first is that for any two integers $k \neq l$, $\int_{-\infty}^{\infty} \phi(t - k)\phi(t - l)dt = 0$ while the second states that for any integer k, $\int_{-\infty}^{\infty} (\phi(t - k))^2 dt = 1$. If we apply the scale invariance property, we find that $\{\phi(2t - k)\}_k$ becomes an orthogonal basis for V_1. Using the same line of reasoning, if we define $\phi_{jk}(t) = 2^{j/2}\phi(2^j t - k)$, then $\{\phi_{jk}(t)\}_k$ forms an orthonormal basis for the space V_j. In referring to ϕ_{jk} as a basis function, we mean that any function f_j which lives in the space V_j can be represented as a sum of these basis functions,

$$f_j(t) = \sum_{k=-\infty}^{\infty} a_{jk}\phi_{jk}(t).$$

Extending this result, we say that any square integrable function f which necessarily lives in $L_2(\mathbb{R})$ can be expanded as a sum of the basis functions ϕ_{jk} across all scales,

$$f(t) = \sum_{k=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} a_{jk}\phi_{jk}(t).$$

The name given to the function ϕ which generates the basis functions for all spaces $\{V_j\}$ is the scaling function of the multiresolution analysis [19], [28], [16], [26].

In order to construct the basis functions for all spaces $\{V_j\}$ from the scaling function ϕ, we look to an equivalence relation known as the Scaling Equation. Its development begins with the following line of reasoning. According to our previous discussion of the nesting of subspaces, $V_0 \subset V_1$ and therefore any function which resides in V_0 can be expanded in terms of the basis functions for V_1. More specifically, the scaling function $\phi \in V_0$ can be written in terms of $\{\phi_{1k}(t)\}_k$,

$$\phi(t) = \sum_{k=-\infty} h_k \phi_{1k}(t) = \sqrt{2} \sum_{k=-\infty} h_k \phi(2t - k). \tag{8}$$

Equation (8) is called the Scaling Equation. This same process can be applied to relate the scaling functions between any two successive levels of resolution,

$$\phi_{J-1,m}(t) = \sqrt{2^{J-1}}\phi\left(2^{J-1}t - m\right) = \sqrt{2^J} \sum_{k=-\infty} h_k \phi(2^J t - k) = \sum_{k=-\infty} h_k \phi_{J,k}(t).$$

The filter coefficients $\{h_k\}$ can be found, using the fact that $\{\phi_{1k}(t)\}$ are orthonormal, by taking the inner product

$$h_k = \langle \phi, \phi_{1k} \rangle = \sqrt{2} \int_{-\infty}^{\infty} \phi(t)\phi(2t - k)dt.$$

Using this process, we are able to find the functions $\phi_{jk}(t)$ which form a basis for any approximation space V_j [28], [19], [16].

We now shift our attention to the detail spaces denoted by $\{W_j\}$. If we use the fact that $\lim_{j \to -\infty} V_j = \{0\}$ and we extend Equation (7) to $\lim_{j \to -\infty}$, we find that we can represent the approximation space V_{j+1} by the orthogonal sum

$$V_{j+1} = \oplus_{k=-\infty}^{j} W_k.$$

If we let $j \to \infty$, we see that

$$L_2(\mathbb{R}) = \oplus_{k=-\infty}^{\infty} W_k.$$

In other words, the set of all square integrable functions can be broken down into orthogonal subspaces, each of which can detect details of the functions up to a given resolution. This means that the union of all the basis functions for $\{W_j\}$ serve in the same manner as a basis for $L_2(\mathbb{R})$. Just as the approximation spaces V_j each had a basis $\{\phi_{jk}(t)\}_k$, every detail space W_j in a multiresolution analysis has its own orthonormal basis $\{\psi_{jk}(t)\}_k$. In similar fashion to $\phi_{jk}(t)$, we define $\psi_{jk}(t) = 2^{j/2}\psi(2^j t - k)$. Thus we may say $L_2(\mathbb{R})$ has an orthonormal basis $\{\psi_{jk}(t)\}_{jk}$ known as the wavelet basis, also referred to as a wavelet family [28], [19], [16], [20].

The function $\psi(t)$ which is used to construct all the basis functions $\psi_{jk}(t)$ for the W_j spaces is known as the mother wavelet. By definition of $\psi_{jk}(t)$ we see that $\{\psi(t - k)\}_k$ belongs to W_0 and since Equation (7) tells us that $W_0 \subset V_1$, $\psi(t)$ can be written as the sum of the basis functions for V_1, $\{\phi_{1k}(t)\}_k$:

$$\psi(t) = \sum_{k=-\infty}^{\infty} g_k \phi_{1k}(t) = \sqrt{2} \sum_{k=-\infty}^{\infty} g_k \phi(2t - k). \tag{9}$$

Equation (9) is known as the Wavelet Equation and it can be extended to find the relationship between the wavelet and the scaling function at the next finer scale for any two successive levels of resolution.

Just as we did to solve for the filter coefficients of the scaling equation, we apply the fact that $\{\phi_{1k}(t)\}_k$ are orthonormal to find the filter coefficients g_k,

$$g_k = \langle \psi, \phi_{1k} \rangle = \sqrt{2} \int_{-\infty}^{\infty} \psi(t)\phi(2t - k)dt.$$

Using this process, we are able to find all of the functions $\psi_{jk}(t)$ needed to form a basis for $L_2(\mathbb{R})$ [19], [28], [16].

The following example will demonstrate how a sample square integrable function can be decomposed into a set of approximations and details occurring at various levels of resolution according to an MRA. Our choice of scaling function and wavelet bases is referred to as Daubechies 4 in [26]. As the specifics on how to construct these bases are described by Walker, who founds his work upon the seminal paper by Ingrid Daubechies [8], we leave the reader to examine these references so as not to draw the discussion too far from the focus of how a multiresolution analysis works. The MATLAB® code written to generate these bases and the ensuing MRA of the sample signal is listed in the Appendices of this paper under the filename $Daub4\_10.m$. For now, let it suffice to say that these scaling functions and wavelets were designed by Daubechies to be orthonormal and of compact support (i.e. the functions only have non-zero value for a finite closed interval).

Figure 7 shows a sample signal which is represented in MATLAB® as a vector of length 1024.

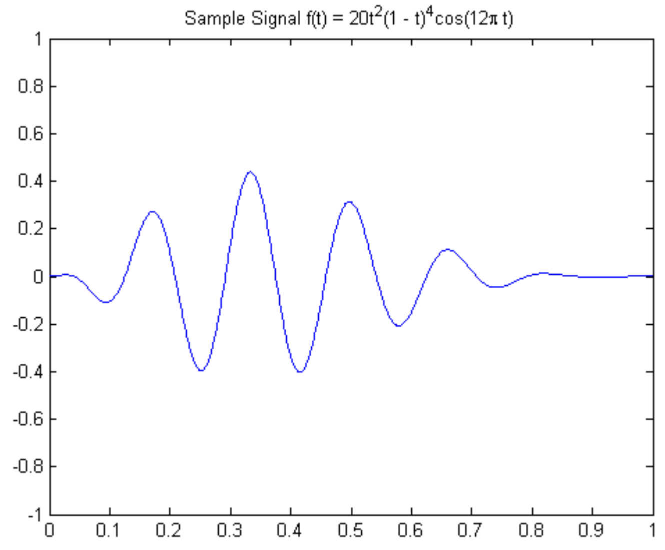

Figure 7: Sample Signal

Since the signal $f(t)$ is of length $1024 = 2^{10}$, the …nest resolution level to which we can decompose it is $j = 10$ and the coarsest level is $j = 1$. The coarse to …ne approximations to $f(t)$, $f_1(t)$ through $f_{10}(t)$, are displayed in Figures 8 and 9.

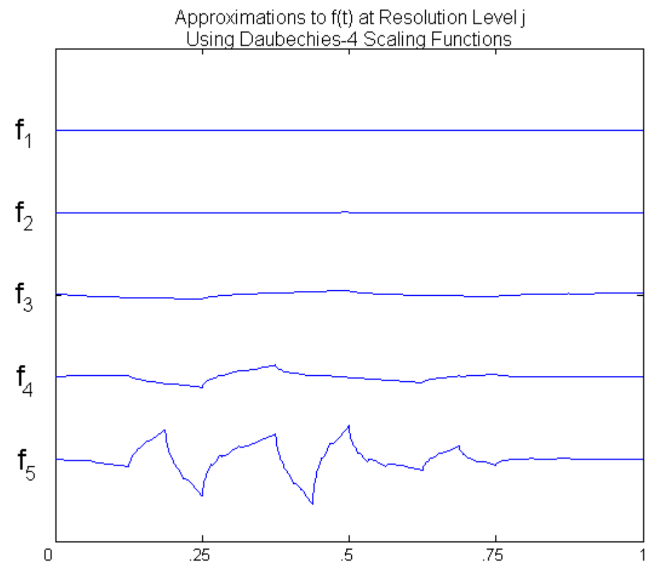

Figure 8: Coarser Approximations to Sample Signal

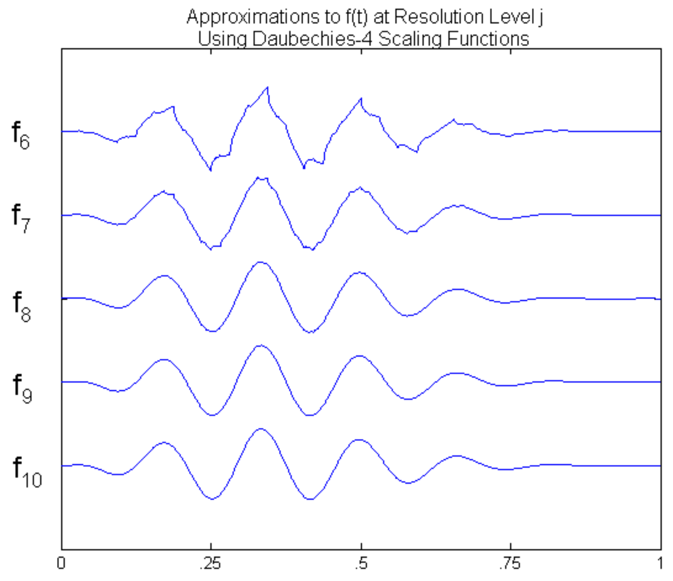

Figure 9: Finer Approximations to Sample Signal

Figures 10 and 11 show the details of $f(t)$, $d_1(t)$ to $d_{10}(t)$, from coarse to fine.

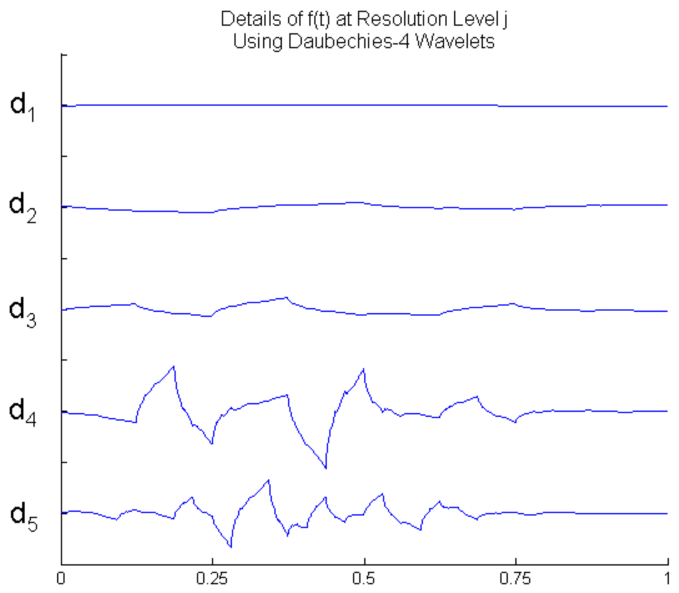

Figure 10: Coarser Details of Sample Signal

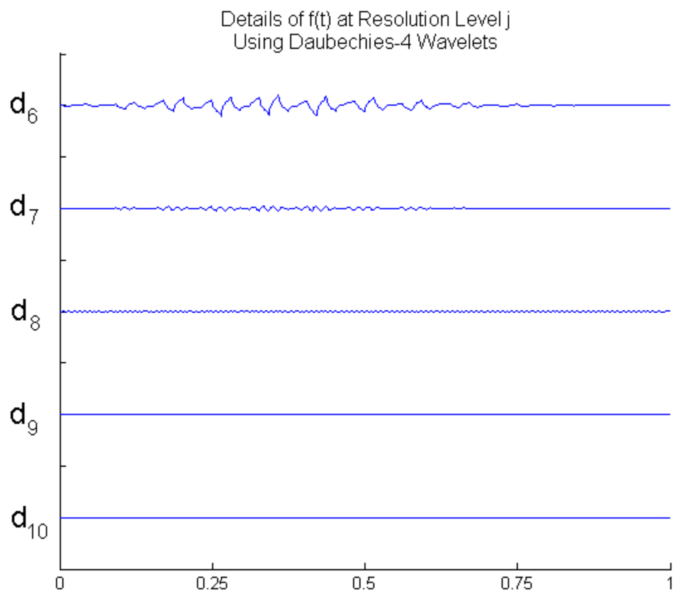

Figure 11: Finer Details of Sample Signal

From examination of Figures 8 and 9, one can see that the …nest approximation of $f(t)$ at resolution level j = 10 does a perfect job of capturing every detail about the signal. This corresponds to the …nest level details of the sample signal, $d_{10}(t)$, taking on the value of zero. In e¤ect, no information is lost when $f(t)$ is approximated by $f_{10}(t)$. As the approximations get progressively coarser, more and more details are left out until we see that the coarsest approximation at level j = 1, $f_1(t)$, only a constant function survives.

We now seek to validate the assertion of Equation (7) and prove that a signal can be perfectly reconstructed by summing up the approximation to it at level j ¡ J with the details from level j down to j ¡ J. We choose the arbitrary value of J = 5 to show a reconstruction after six levels of decomposition. Figure 12 demonstrates the fact that

$$f(t) = d_{10}(t) + d_9(t) + d_8(t) + d_7(t) + d_6(t) + d_5(t) + f_5(t).$$

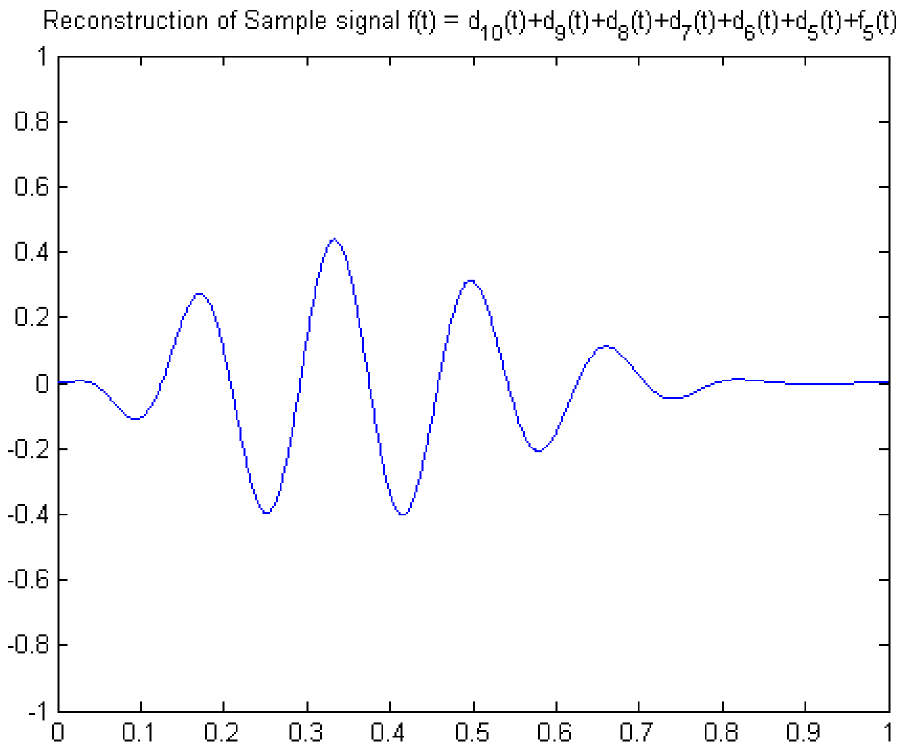

Figure 12: Reconstruction of Sample Signal

So far, our discussion of wavelets and multiresolution analysis has taken place in the context of one-dimensional function spaces. The problem of optic ‡ow, however, involves two-dimensional images. Therefore, we must see how these concepts evolve in 2-D.

To construct a two-dimensional scaling function, one merely takes the tensor product of two one-dimensional scaling functions,

$$\phi(x_1, x_2) = \phi(x_1)\phi(x_2).$$

In turn, the Scaling Equation becomes

$$\phi(x_1, x_2) = 2 \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} h_{k,l}\phi(2x_1 - k, 2x_2 - l). \tag{10}$$

Both $\phi(x_1)$ and $\phi(x_2)$ satisfy the 1-D Scaling Equation $\phi(x) = \sqrt{2}\sum_{k=-\infty}^{\infty} h_k\phi(2x - k)$ so therefore we may write $h_{k,l} = h_k h_l$. Thus, the 2-D Scaling Equation is the product of two 1-D Scaling Equations [28].

We now approach a shift in terminology for the two-dimensional case. Here, a 2-D Mother Wavelet can be any of four combinations denoted by,

$$\theta^1(x_1, x_2) = \psi(x_1)\phi(x_2),$$
$$\theta^2(x_1, x_2) = \phi(x_1)\psi(x_2),$$
$$\theta^3(x_1, x_2) = \psi(x_1)\psi(x_2),$$
$$\theta^4(x_1, x_2) = \phi(x_1)\phi(x_2).$$

Notice that the general 2-D Mother Wavelet $\theta^n(x_1, x_2)$, with index $n = 1, 2, 3,$ *or* 4, can be composed of a tensor product of either 1-D scaling functions, wavelets, or both. The significance of each lies in the different types of information they provide. θ^1 is known as the High-Low component because it provides detailed high-resolution information in the x_1 direction and long-term trend information in the x_2 direction. This stems from the fact that 1-D wavelets form bases for the detail spaces of a signal's decomposition while 1-D scaling functions form bases for the approximation or "trend" spaces. Thinking of an image as a 2-D signal, θ^1 would serve best to capture vertical lines appearing in the image. Similarly θ^2, being the Low-High component, would provide the most information about horizontal lines occurring in an image [3, Ch. 4], [28].

At this point, the basic theory behind wavelets and multiresolution analysis has been explained. We now seek to demonstrate how Bernard makes use of these concepts to solve the problem of optic flow.

6.3 Application of Wavelets to Optic Flow

A key question still remains unanswered. What is the exact purpose wavelets serve in solving for the coefficients of the optic flow vectors v_1 and v_2? The answer lies in the multiscale nature of wavelets.

In real life, actual motion of an object happens over a continuum of both space and time. This is not the case in the context of an image sequence. In this digital environment, motion is represented by an object comprising a set number of pixels that shifts over a certain number of pixels in the x_1 and x_2 directions. It may help to think of the space-time continuum being projected onto a discrete grid of pixels and frames. This is exactly why wavelets become so beneficial.

We define the 2-D wavelet family

$$\theta^n_{j\mathbf{k}}(\mathbf{x}) = 2^{j/2}\theta^n\left(2^j\mathbf{x} - \mathbf{k}\right) \quad where\ j \in \mathbb{Z}\ \ and\ \ \mathbf{k} \in \mathbb{Z}^2\ and\ n = 1, \ldots, 4.$$

Here Z represents the set of all integers and Z^2 the 2-D Cartesian grid of integers. Let us consider a video sequence where every image is of a 2^J by 2^J pixel grid. Standard practice is to think of this area being normalized so that it has length 1 in both the x_1 and the x_2 direction. Accordingly, each pixel is considered to be a square with each of its sides having length $\frac{1}{2^J} = 2^{¡J}$. At each resolution level j, the 2-D wavelet cannot detect details of the image which are of scale less than $2^{¡j}$. In other words, $\theta^n_{j\mathsf{k}}(\mathsf{x})$ can only detect features of an image which are of size $2^{J¡j}$ pixels by $2^{J¡j}$ pixels or larger. The wavelets at the …nest resolution level J, namely $\theta^n_{J\mathsf{k}}(\mathsf{x})$, can therefore zoom in on details which are only one pixel wide by one pixel long [3], [7].

From this result we are able to see how a multiresolution analysis performed with a two-dimensional wavelet basis allows us to capture both large and small scale motion occurring in an image sequence at the same time. An object which is moving at a relatively high velocity will be represented as jumping across large gaps in the pixel grid from frame to frame. Wavelets at the coarser levels of resolution have a larger support and so they will be able to detect this change in location of the object. Conversely, wavelets at …ner levels of resolution do not have a support large enough to enclose the entire displacement travelled by the object and so they cannot measure its motion within their window. This is the aperture problem. On the other hand, an object travelling at a relatively low velocity will only move across a few pixels in the grid from frame to frame. Wavelets at the …ner levels of resolution will have support small enough to zoom in and detect the fact that the object may have only shifted over by a few pixels. For those wavelets at the coarser levels of resolution, however, the motion may not represent a large enough trend and so it will suppressed as too small of a detail. This is what is known as time aliasing.

We see, however, that the time aliasing vs. aperture problem is avoided when we simultaneously apply all levels of resolution in our MRA to the motion occurring between two frames. The large scale motion is represented in the coarser levels of resolution. This information will translate into long vector arrows in the optic ‡ow …eld. Small scale motion can be seen at the …ner levels of resolution and will translate into shorter vector arrows in the solution for the optic ‡ow. Additionally, the application of the di¤erent components of the 2-D wavelets, namely θ^1,θ^2,θ^3, and θ^4, to the image sequence allows us to account for the detection of an object's motion in any direction [3], [4], [5], [6], [7].

In theory, everything seems to work out. Let us now turn to Bernard's method of solving for the optic ‡ow vectors to see if these wavelets really are overcoming the challenges of the problem when put into practice. Keep in mind throughout the discussion that our end goal is to solve for a …nite number of vectors, each of which will be applied to a certain subgrid of pixels in the image to represent the optic ‡ow.

Once again we begin with the governing equation for optic ‡ow,

$$\frac{\partial I}{\partial x_1} v_1(\mathsf{x}) + \frac{\partial I}{\partial x_2} v_2(\mathsf{x}) + \frac{\partial I}{\partial t} = 0.$$

We hit the governing equation with a basis function $\theta^n_{\mathsf{u}}(x) = \theta^n(\mathsf{x} ¡ \mathsf{u})$ de…ned in our notation earlier, and then integrate,

$$\iint \left(\frac{\partial I}{\partial x_1} v_1(\mathbf{x}) + \frac{\partial I}{\partial x_2} v_2(\mathbf{x}) + \frac{\partial I}{\partial t} \right) \theta^n(\mathbf{x} - \mathbf{u}) dx_1 dx_2 = 0, \qquad for\ n = 1, .., N \ . \tag{11}$$

Here N could equal 3 or 4. Equation (11) can be expanded and rewritten in inner product notation,

$$\left\langle \frac{\partial I}{\partial x_1} v_1, \theta^n_{\mathbf{u}} \right\rangle + \left\langle \frac{\partial I}{\partial x_2} v_2, \theta^n_{\mathbf{u}} \right\rangle + \left\langle \frac{\partial I}{\partial t}, \theta^n_{\mathbf{u}} \right\rangle = 0, \qquad for\ n = 1, .., N \ . \tag{12}$$

At this point, Bernard makes his most critical assumption. He states that the optic flow vectors $v_1(\mathbf{x})$ and $v_2(\mathbf{x})$ are constant over the supports of the basis functions θ^n_u , i.e.

$$v_1(\mathbf{x}) = v_1(\mathbf{u}) \quad for\ all\ \mathbf{x} \in \text{support } \theta^n_u, \quad for\ all\ n,$$
$$v_2(\mathbf{x}) = v_2(\mathbf{u}) \quad for\ all\ \mathbf{x} \in \text{support } \theta^n_u, \quad for\ all\ n.$$

Therefore, if the optic flow vectors are constant over the domain of the inner products, they can be taken outside of the integrals. In addition, since the inner products are integrals being taken with respect to x, we can take the partial derivative with respect to time $\frac{\partial}{\partial t}$ outside of the third inner product term,

$$\left\langle \frac{\partial I}{\partial x_1}, \theta^n_{\mathbf{u}} \right\rangle v_1(\mathbf{u}) + \left\langle \frac{\partial I}{\partial x_2}, \theta^n_{\mathbf{u}} \right\rangle v_2(\mathbf{u}) + \frac{\partial}{\partial t} \langle I, \theta^n_{\mathbf{u}} \rangle = 0, \qquad for\ n = 1, .., N \ . \tag{13}$$

If we perform an integration by parts for the first two inner products, we arrive at the equation

$$\left\langle I, \frac{\partial \theta^n_{\mathbf{u}}}{\partial x_1} \right\rangle v_1(\mathbf{u}) + \left\langle I, \frac{\partial \theta^n_{\mathbf{u}}}{\partial x_2} \right\rangle v_2(\mathbf{u}) = \frac{\partial}{\partial t} \langle I, \theta^n_{\mathbf{u}} \rangle, \qquad for\ n = 1, .., N \ . \tag{14}$$

Note: The terms $\left. I\theta^n_{\mathbf{u}} \right|_b^a$ are equal to 0 and therefore, they do not appear in Equation (14). This is because a and b represent the boundaries of the image and the basis functions $\theta^n_{\mathbf{u}}$ only have finite support so they vanish on the boundaries.

Equation (14) is a key equation because if we now turn back to $\theta^n_{\mathbf{u}}$ for $n = 1, .., N$ where $N = 3$ or 4 as defined previously in our notation, we have a projected system of N equations with two unknowns $v_1(\mathbf{u})$ and $v_2(\mathbf{u})$,

$$\begin{cases} \left\langle I, \frac{\partial \theta^1_{\mathbf{u}}}{\partial x_1} \right\rangle v_1(\mathbf{u}) + \left\langle I, \frac{\partial \theta^1_{\mathbf{u}}}{\partial x_2} \right\rangle v_2(\mathbf{u}) = \frac{\partial}{\partial t} \langle I, \theta^1_{\mathbf{u}} \rangle \\ \vdots \qquad \vdots \qquad \vdots \\ \left\langle I, \frac{\partial \theta^N_{\mathbf{u}}}{\partial x_1} \right\rangle v_1(\mathbf{u}) + \left\langle I, \frac{\partial \theta^N_{\mathbf{u}}}{\partial x_2} \right\rangle v_2(\mathbf{u}) = \frac{\partial}{\partial t} \left\langle I, \theta^N_{\mathbf{u}} \right\rangle \end{cases} . \tag{S}$$

We say that this system, which Bernard refers to as (S), is projected because for each u that we could choose in translating across the image, we are able to say that the optic ‡ow vector v is constant over the support of θ_u^n [3]. The problem with this, however, is that u lives in the continuum so therefore our individual optic ‡ow vectors could be constant over a certain area of the image but not necessarily for a certain group of pixels. This leaves us with the fact that we still have a vector …eld v we are solving for and not a …nite system of individual vectors as we desire.

Furthermore, there is no mention of scale in (S). We need to …t in some way of re…ning this scheme so that we can account for average trends and di¤erences over various groupings of pixels. This insertion of multiresolution analysis will allow our discrete set of optic ‡ow vectors more closely approximate the continuous optic ‡ow …eld [3].

Finally, we must also contend with the issue that an image sequence is sampled in time (i.e. motion is captured with a discrete number of image frames). Therefore, the right hand side of (S) which is a partial derivative with respect to time,

$$\frac{\partial}{\partial t}\langle I, \theta_{\mathrm{u}}^n\rangle\,,$$

must be approximated with some sort of …nite di¤erence such as:

$$\frac{\partial I}{\partial t} \simeq I(t+1) - I(t),$$

where $t+1$ really just represents the next image frame from the one at time t.

In his thesis [3], Bernard actually uses a higher order estimate to the time derivative and measures the optic ‡ow at each $t + 1/2$ according to

$$\frac{\partial I(t+1/2)}{\partial t} \simeq I(t+1) - I(t). \tag{15}$$

This approximation makes sense if one considers the Fundamental Theorem of Calculus for derivatives: $f'(c)(b-a) = f(b) - f(a)$ where $b - a = 1$ for our example.

Now we will introduce di¤erent scalings of the basis functions so that we may perform an MRA later on to solve for our optic ‡ow vector:

$$\theta_{\mathrm{u}s}^n(\mathrm{x}) = s^{-1}\theta^n\left(\frac{\mathrm{x}-\mathrm{u}}{s}\right),$$

where s is a continuous scaling index. We examine the inner products on the left hand side of (S) evaluated at $t + 1/2$,

$$\left\langle I(t+1/2), \frac{\partial}{\partial x_i}\theta_{\mathrm{u}s}^n\right\rangle. \tag{16}$$

We use the following averaging estimation to approximate I at a non-integer time (i.e. in between frames):

$$I(t+1/2) \approx \frac{I(t)+I(t+1)}{2}. \tag{17}$$

If we make use of (15), (16), and (17) we can form a new projected system of equations that includes various scales of the basis functions and also is discretized in time,

$$\left\{\begin{array}{l} \sum_{i=1,2} \left\langle \frac{I(t+1)+I(t)}{2}, \frac{\partial \theta^1_{\mathrm{u}s}}{\partial x_i} \right\rangle v_i(\mathrm{u}) = \left\langle I(t+1) - I(t), \theta^1_{\mathrm{u}} \right\rangle \\ \qquad\qquad \vdots \qquad\qquad\qquad\qquad \vdots \qquad\quad \vdots \\ \sum_{i=1,2} \left\langle \frac{I(t+1)+I(t)}{2}, \frac{\partial \theta^N_{\mathrm{u}s}}{\partial x_i} \right\rangle v_i(\mathrm{u}) = \left\langle I(t+1) - I(t), \theta^N_{\mathrm{u}} \right\rangle \end{array}\right. . \tag{DS}$$

One problem still remains, we are still trying to solving for our optic flow as a vector field rather than a set of individual vectors [3], [5], [6], [7]. This is because in (DS) , we are still employing the use of a continuous translation parameter u and a continuous scaling/dilation parameter s . In order to resolve this issue, we sample the continuous parameters u and s to create discrete parameters k and j . We formulate a new set of basis functions which Bernard now begins calling a wavelet family $\{\theta^n_{j\mathrm{k}}\}_{n=1,..,N;\, j\in Z; \mathrm{k}\in Z^2}$ defined by:

$$\theta^n_{j\mathrm{k}}(\mathrm{x}) = 2^{j/2}\theta^n(2^j\mathrm{x} - \mathrm{k}),$$

where j is a resolution index, and $\mathrm{k} = (k_1, k_2)$ is a 2-D translational index. The 2-D wavelet $\theta^n_{j\mathrm{k}}$ is located around $(2^{-j}k_1, 2^{-j}k_2)$ and has support over a domain of size proportional to 2^{-j} .

Using this new set of discrete translation indices, we may now express our system of equations as:

$$\left\{\begin{array}{l} \left\langle I, \frac{\partial \theta^1_{j\mathrm{k}}}{\partial x_1} \right\rangle v_1 + \left\langle I, \frac{\partial \theta^1_{j\mathrm{k}}}{\partial x_2} \right\rangle v_2 = \left(\frac{\partial I}{\partial t}, \theta^1_{j\mathrm{k}} \right) \\ \qquad\qquad \vdots \qquad\qquad\qquad\quad \vdots \qquad\quad \vdots \\ \left\langle I, \frac{\partial \theta^N_{j\mathrm{k}}}{\partial x_1} \right\rangle v_1 + \left\langle I, \frac{\partial \theta^N_{j\mathrm{k}}}{\partial x_2} \right\rangle v_2 = \left(\frac{\partial I}{\partial t}, \theta^N_{j\mathrm{k}} \right) \end{array}\right. . \tag{$S_{j\mathrm{k}}$}$$

The approximations for the discretization of time used in (DS) are left out in $(S_{j\mathrm{k}})$ merely to reduce clutter in the equations and allow the reader to see how we now have a finite system of equations (either 3 or 4) for which we must solve for 2 variables v_1 and v_2 for each translational and resolutional combination j, k . The finite difference approximations to the time derivatives used in (DS) are actually used for performing the calculations.

As such, solving for the optic ‡ow has gone from being an ill-posed problem to one that is well-posed. Thus, we now have a matrix manipulation problem to solve for a …nite number of vectors since j and k are …nite [3], [7]:

$$\boxed{M_{j\mathrm{k}}\mathrm{v} = Y_{j\mathrm{k}}} \tag{18}$$

Seemingly, all of our obstacles have been overcome in transitioning to the …nite system of di¤erence equations in Equation (18). The trick to seeing that the solutions to the problem in Equation (18) will be stable (i.e. the time-aliasing vs. aperture problem has been surmounted) comes from the fact that the velocity vectors $\mathrm{v}(\mathrm{x}, t)$ are being solved for at each resolution level j in the multiresolution analysis. The determinant of $M_{j\mathrm{k}}$ could be 0 or very close to it if the motion is occurring on a scale much di¤erent than 2^{-j} in that region covered by the support of $\theta^n_{j\mathrm{k}}$. This would cause the approximation to v to be very unstable at that level of resolution. However, due to the fact that Bernard uses two indices, j and k, to achieve a sampling which is spatially localized at various scales, the rate of approximation for a given region is unhindered by a particular resolution level yielding an unstable result. In other words, since there can only be one vector assigned to represent the optic ‡ow at location x at time t, the solutions from each level of resolution collectively represent enough information to describe whatever motion is occurring, either large or small scale, with one single vector. Thus we see that Bernard's approach can be viewed mathematically as an attempt to stabilize the approximation to v more so than as a way to regularize the problem [3], [7].

Issues of stability with regards to solving an overdetermined system, as Equation (18) plainly is, are not addressed within the scope of this paper, however, they are dealt with brie‡y by Bernard in [3, Ch. 4.1.4]. The …nal challenge to implementing Bernard's method of solving for optic ‡ow stems from having to take the derivative of a two-dimensional wavelet as is required to set up the system of equations $S_{j\mathrm{k}}$.

6.4 Obstacle to Implementation

Although the computation of the inner products $\left(I, \dfrac{\partial \theta^n_{j\mathrm{k}}}{\partial x_1} \right)$ and $\left(I, \dfrac{\partial \theta^n_{j\mathrm{k}}}{\partial x_2} \right)$ may not seem overly complicated at …rst glance, a signi…cant obstacle lies in the requirement to take the derivative of a two-dimensional wavelet. Here the di¢culty arises from the need to construct wavelets in a manner that lends itself to taking their derivative. Our contention is that we need not construct the derivative at every point over R^2. Rather, when dealing with a discrete signal I, the value of $\dfrac{\partial \theta^n_{j\mathrm{k}}}{\partial x_1}$ and $\dfrac{\partial \theta^n_{j\mathrm{k}}}{\partial x_2}$ need only be known at the integer points of (x_1, x_2) [22].

The discussion of this theory [22] will be conducted in one-dimension so as to avoid the confusion and clutter accompanying the analogous explanation in two-dimensions. Our example problem begins with $g(t)$, a 1-D analog signal which has compact support $[0, L]$ where $L = 2^J$ for some number J and $g(L) = 0$. The discrete approximation to $g(t)$ is formed so that the sampling interval is of length 1, $b(n) = g(n)$. To ease the burden of notation, we employ the normalized signal $f(t)$ having support $0 \le t \le 1$ such that $f(t) = g(2^J t)$. Therefore, we see that $f(t)$ is sampled at intervals of length $2^{-J} = \frac{1}{L}$ so that $f\left(\frac{n}{2^J}\right) = b(n)$.

Analysis of the signal will be conducted using a dyadic, multi-resolutional decomposition of $L_2(\mathrm{R})$ based upon a scaling function ϕ of compact support $[0, N]$ with $\phi(N) = 0$. By dyadic, we mean that each level of resolution drops down in coarseness by a factor of 2. As is stated in [19], ϕ satisfies the scaling equation built from a discrete filter $\{h_n\}$ having support $0 \le n \le N - 1$,

$$\phi(t) = \sqrt{2} \sum_{l=0}^{N-1} h_l \phi(2t - l).$$

In the multi-resolutional decomposition of $L_2(\mathrm{R})$ based upon ϕ, the function $f(t)$ is said to belong to the space V_J where $2^{-J} = \frac{1}{L}$ is the sampling interval. The goal of our approach is to initialize the multiresolution by approximating the values of $\left\{ \left\langle f, \phi_{J,k} \right\rangle \right\}_{k=\min}^{\max}$ which are indexed over a range $\min \le k \le \max$ of non-zero values, and then to engage a cascade algorithm which is based upon the scaling equation to iteratively find the inner products $\left\{ \left\langle f, \phi_{(J-j),k} \right\rangle \right\}$ for $1 \le j \le J$.

From [8], the scaling equation can be evaluated for an integer value according to

$$\begin{aligned} \phi(n) &= \sqrt{2} \sum_{l=0}^{N-1} h_l \phi(2n - l) \\ &= \sqrt{2} \sum_{k} h_{k-2n} \phi(k). \end{aligned}$$

where k is indexed by $0 \le k - 2n \le N - 1$.

Defining the matrix

$$m'_{n,k} = \begin{cases} \sqrt{2} h_{k-2n} & 0 \le k - 2n \le N - 1 \\ 0 & otherwise \end{cases},$$

and restricting to the integer values, the scaling equation can be rewritten

$$\phi(n) = \sum_{k} m'_{n,k} \phi(k) \qquad where \;\; 0 \le k - 2n \le N - 1. \tag{19}$$

Differentiating the scaling equation, a new scaling equation for ϕ' results,

$$\phi'(t) = 2\sqrt{2} \sum_{l=0}^{N-1} h_l \phi'(2t - l). \tag{20}$$

If we restrict Equation (20) to the integers and apply the principle of substitution from Equation (19) we obtain

$$\left(\frac{1}{2}\right) \phi'(n) = \sum_{k} m'_{n,k} \phi'(k) \qquad where \;\; 0 \le k - 2n \le N - 1. \tag{21}$$

From Equations (19) and (21), we see that the sequence of integer-sampled values of the scaling function ϕ comprise a right eigenvector for the matrix $m0$ with eigenvalue 1, and the integer-sampled values of the derivative of the scaling function ϕ' make up a right eigenvector for $m0$ with eigenvalue $1/2$. In [8], a rather cryptic process unfolds in order to find which eigenvectors correspond to the sampling of the scaling function and the sampling of its derivative. The exact code required to perform this process in MATLAB® is included in the Appendix. In short, the first part of Daubechies' scheme chooses that $\lambda = 1$ eigenvector which is orthogonal to the left eigenvector $[1, 1, 1, 1]$ of $m0$ whose sum of components is equal to 1. This will be the sequence $\{\phi(n)\}$ where $0 \le n \le N - 1$ which are the non-zero values of the integer samplings of the scaling function. The second part of the scheme chooses that $\lambda = 1/2$ eigenvector whose sum with the vector $[1, 2, 3, 4]$ belongs to the span of the previous left eigenvector $[1, 1, 1, 1]$. This will in turn be the sequence $\{\phi'(n)\}$ where $0 \le n \le N - 1$ which are the non-zero values of the integer samplings of the derivative of the scaling function [8].

Having found the sequence $\{\phi'(n)\}_{n=0,\ldots,N-1}$, we may initialize the algorithm for computing inner products of the signal with the derivative of the scaling function. We continue to use J as the dyadic exponent determined by the support of the original signal $g(t)$, so that $2^J = L$. In addition, we make use of the definition $\phi'_{J,m}(t) = 2^{J/2}\phi'(2^J t - m)$. For any integer value of m, we can approximate the value of the inner product $\left\langle f, \left(\phi_{J,m}\right)' \right\rangle$ according to:

$$\left\langle f, \left(\phi_{J,m}\right)' \right\rangle = \int_{t=-\infty}^{\infty} f(t) \cdot \frac{d}{dt} \left(\sqrt{2^J}\phi(2^J t - m) \right) dt,$$

using the Chain Rule and the fact that $0 \le t \le 1$,

$$= \sqrt{2^J}2^J \int_{t=0}^{1} f(t)\phi'(2^J t - m)dt,$$

breaking up the integral into segments,

$$= \sqrt{2^J}2^J \sum_{n=0}^{2^J-1} \int_{t=n/2^J}^{(n+1)/2^J} f(t)\phi'(2^J t - m)dt,$$

using a Riemann approximation to the segments of the integral,

$$\cong \sqrt{2^J}2^J \sum_{n=0}^{2^J-1} f\left(n/2^J\right) \cdot \phi'(2^J\left(n/2^J\right) - m) \cdot \left(1/2^J\right),$$

cancelling out the 2^J factors and substituting $f\left(n/2^J\right) = b(n)$,

$$\cong \sqrt{2^J} \sum_{n=0}^{2^J-1} b(n) \cdot \phi'(n - m),$$

making use of the convolution operator $*$,

$$= \sqrt{2^J} \cdot \left(\underline{b} * \underline{\tilde{\phi}}' \right)(m),$$

where

$$\underline{b} = \{b(n)\}_{n=0}^{n=L-1-2^J-1}$$

$$\underline{\phi}' = \left\{ \phi'(n) \right\}_{n=0}^{n=N-1}$$

$$\underline{\tilde{\phi}}'(n) = \underline{\phi}'(-n).$$

Thus the inner product of our signal with the derivative of the scaling function has been reduced to a seemingly simple convolution between the integer-sampling points of these two functions. However, one issue still remains before we can begin computing these coefficients, that being the boundary conditions of the signal. Obviously, a translated and dilated derivative scaling function will have different boundaries than the original ϕ'. This means that their support will extend past the boundaries of the signal, drastically affecting the corresponding inner products, a fact that cannot be ignored [22].

The basis for addressing this problem is presented in [19, Ch. 7.5]. He contends that in a case such as this, one must employ a circular convolution. The idea is to extend the "discrete signal" $\underline{b}$ to make it a periodic signal with period L, and the "discrete filter" $\underline{\phi}'$ to a periodic filter with period max(N, L). In essence the convolution becomes a "circular convolution" of two periodic signals that has itself a fundamental period L. From this information, we extract the "fundamental period signal", a sequence indexed by $0 \leq k \leq N-1$. This sequence represents what we were seeking, the inner products at resolution level J [24], [22].

Within the WAVELAB® add-on package to MATLAB®, written by Dr. David Donoho and his colleagues at Stanford University, there exists an m-file aconv.m which performs the circular convolution according to:

$$\text{J Level Inner Products} = \sqrt{2^J} \cdot aconv(Dphi0, b).$$

where $Dphi0$ represent the integer samplings of the derivative of the scaling function, $\{\phi'(n)\}_{n=0,\ldots,N-1}$, and b is the vector containing the integer samplings of the original function $g(t)$ [22].

Having found the inner products at level J, denoted from now on as JLevInnerProds, we seek now to find the inner products at resolution level $J-1$, denoted by J_1LevInnerProds. We make use of the scaling equation as it applies to any two successive levels in a multiresolution [19],

$$\phi_{(J-1),\,m}(t) = \sum_{l=0}^{N-1} h_{l-2n} \phi_{J,\,l}(t), \tag{22}$$

evoking once more the definition $\phi_{j,k}(t) = 2^{j/2}\phi(2^j t - m)$. We differentiate Equation (22) to yield

$$\left(\phi_{(J-1),\,m}\right)'(t) = \sum_{l=0}^{N-1} h_{l-2n}\left(\phi_{J,\,l}\right)'(t). \qquad (23)$$

Note that the derivative in Equation (23) is being taken after the scaling function ϕ is dilated and translated, i.e. $\left(\phi_{j,k}\right)'$ not $\left(\phi'\right)_{j,k}$. Substituting this result into the inner product we see

$$\left\langle f,\ \left(\phi_{(J-1),\,m}\right)'\right\rangle = \sum_{l=0}^{N-1} h_{l-2n}\left\langle f,\ \left(\phi_{J,\,l}\right)'\right\rangle. \qquad (24)$$

Here, let $\underline{h}$ denote the sequence of non-zero coefficients for the filter $\underline{h}(k) = h_k$ indexed over the range $0 \le k \le N-1$. Further, let $\tilde{\underline{h}}$ denote the filter under involution, i.e. $\tilde{\underline{h}}(k) = \underline{h}(-k)$ which is a sequence indexed over $-(N-1) \le k \le 0$. We apply our finding of JLevInnerProds to rewrite Equation (24) in terms of a convolution as outlined in [19],

$$\text{J\_1LevInnerProds} = \left\langle f,\ \left(\phi_{(J-1),\,m}\right)'\right\rangle = \left(\text{JLevInnerProds} \star \tilde{\underline{h}}\right)(2m). \qquad (25)$$

Notice that after the convolution is performed, the result is downsampled by a factor of 2 according to the argument $2m$. This dyadic decimation of the inner products corresponds to the coarsening of the resolution by a factor of 2 as one goes from level J to level $J-1$.

The issue of the signal's boundary must also be remembered when computing these lower resolution inner products. Fortunately, the WAVELAB® m-file DownDyadLo.m accounts for this fact in addition to performing the required downsampling. Therefore, we can efficiently engage our cascade algorithm for finding the $J-1$ level inner products by

$$\text{J\_1LevInnerProds} = DownDyadLo(\text{JLevInnerProds}, h), \qquad (26)$$

where h represents the vector containing the values $\underline{h}(k) = h_k$ indexed over the range $0 \le k \le N-1$.

The cascade algorithm based upon the scaling equation extends further down the levels of resolution as well,

$$\text{J\_2LevInner Prods} = \left\langle f,\ \left(\phi_{(J-2),\,m}\right)'\right\rangle = \left(\text{J\_1LevInnerProds} \star \tilde{\underline{h}}\right)(2m), \qquad (27)$$

and is implemented in WAVELAB® by

$$\text{J\_2LevInner Prods} = DownDyadLo(\text{J\_1LevInnerProds}, h). \qquad (28)$$

According to the dyadic decimation of the inner products, this process of performing a circular convolution and then downsampling allows the multiresolution to extend down J levels, i.e. to the $J - J = 0$ level of resolution, where there will only be 1 inner product being computed for the set J_JLevInnerProds [22].

Throughout this discussion, we have been using the scaling function ϕ to generate a basis for our multiresolution analysis. What if we had desired to use the mother wavelet ψ instead? Looking to the wavelet equation, we see that the answer is not overly complicated. Recall,

$$\psi(t) = \sqrt{2}\sum_{l=0}^{N-1} g_l \phi(2t - l),$$

where the filter coefficients g_l can be solved for $g_l = \sqrt{2}\int_{-\infty}^{\infty} \psi(t)\phi(2t - l)dt$.
We restrict the wavelet equation to the integer values as before in the scaling equation,

$$\psi(n) = \sqrt{2}\sum_{l=0}^{N-1} g_l \phi(2n - l),$$

$$= \sqrt{2}\sum_{k} g_{k-2n}\phi(k),$$

where $0 \le k - 2n \le N - 1$. Now if we define the matrix $m1$

$$m1_{n,k} = \begin{cases} \sqrt{2}g_{k-2n} & 0 \le k - 2n \le N - 1 \\ 0 & otherwise \end{cases},$$

we may rewrite the wavelet equation restricted to the integers as

$$\psi(n) = \sum_{k} m1_{n,k}\phi(k) \qquad where \;\; 0 \le k - 2n \le N - 1.$$

Now, by applying the results of Equations (19) to (21), we see that we may find the exact values of the derivative of the mother wavelet at the integers according to the matrix multiplication

$$\psi'(n) = 2\sum_{k} m1_{n,k}\phi'(k) \qquad where \;\; 0 \le k - 2n \le N - 1.$$

We now seek to find the inner products of the signal with these values $\{\psi'(n)\}_{n=0,..,N-1}$. This can be accomplished using the same process as outlined for finding the J level inner products of ϕ'. Therefore, the circular convolution is performed to find

$$\text{Wavelet J Level Inner Products} = \left\langle f, \left(\psi_{J,\,m}\right)' \right\rangle = \sqrt{2^J} \cdot aconv(Dpsi0, b),$$

where $Dpsi0$ represents the integer samplings of the derivative of the mother wavelet, $\{\psi'(n)\}_{n=0,..,N-1}$, and b is the vector containing the integer samplings of the original function $g(t)$ [22].

Just as in Equation (22), we can extend the wavelet equation to any two successive levels in the multiresolution [19],

$$\psi_{(J-1),\,m}(t) = \sum_{l=0}^{N-1} g_{l-2n}\phi_{J,\,l}(t), \tag{29}$$

again using the definition $\psi_{j,k}(t) = 2^{j/2}\psi(2^j t - m)$ [19]. We apply the results of Equations (23) and (24) to obtain the equation for inner products with wavelets at lower levels of resolution,

$$\left\langle f,\ \left(\psi_{(J-1),\,m}\right)'\right\rangle = \sum_{l=0}^{N-1} g_{l-2n}\left\langle f,\ \left(\phi_{J,\,l}\right)'\right\rangle.$$

Since ψ is built from ϕ as in Equation (29), the exact same initialization to the cascade algorithm applies. The integer samplings of the derivative of the scaling function, $\left\{\phi'(n)\right\}_{n=0}^{N-1}$, are used to find the inner products at level J, $\left\langle f,\ \left(\phi_{J,\,m}\right)'\right\rangle$, what we have named JLevInnerProds. These are used by the m-file DownDyadLo.m to compute the $J-1$ level inner products with the scaling function, $\left\langle f,\ \left(\phi_{(J-1),\,m}\right)'\right\rangle$. However, WAVELAB®'s matching m-file DownDyadHi.m uses them also to compute the $J-1$ level inner products with wavelets, $\left\langle f,\ \left(\psi_{(J-1),\,m}\right)'\right\rangle$. Within the process of evaluating

$$\text{WavJ\_1LevInnerProds} = \left\langle f,\ \left(\psi_{(J-1),\,m}\right)'\right\rangle = DownDyadHi(\text{JLevInnerProds}, h), \tag{30}$$

DownDyadHi.m performs the exact same tasks (i.e. circular convolution, downsampling by a factor of 2) as DownDyadLo.m, however, it manipulates the filter h to formulate the filter g according to $g_k = (-1)^k h_{1-k}$. Therefore, just as was the case with the scaling functions, the inner products can be found with the derivative of the wavelets to extend the multiresolution down J levels to the $J - J = 0$ level of resolution, where the dyadic decimation will have left only 1 inner product to be computed for the set WavJ_JLevInnerProds [22].

Finally, as we have seen that a multiresolution can be carried out iteratively using the derivative of both a 1-D scaling function and a 1-D wavelet, we recall that we must compute the inner products between the derivatives of our two-dimensional wavelets and the grayscale intensity function, $\left\langle I, \frac{\partial\theta^n_{jk}}{\partial x_1}\right\rangle$ and $\left\langle I, \frac{\partial\theta^n_{jk}}{\partial x_2}\right\rangle$, to solve Bernard's equations S_{jk}. The process we use to generate these inner products relies on the same concept of separability that was applied to create the two-dimensional wavelets θ^n (for $n = 1,..,4$) from the tensor products of one-dimensional scaling functions and wavelets, ϕ and ψ [3], [7], [28].

For example, let's say we seek to find the inner products $\left\langle I, \frac{\partial\theta^1_{jk}}{\partial x_1}\right\rangle$ and $\left\langle I, \frac{\partial\theta^1_{jk}}{\partial x_2}\right\rangle$. Since $\theta^1_{jk}(x_1, x_2) = \psi_{jk_1}(x_1)\phi_{jk_2}(x_2)$, we see that $\frac{\partial\theta^1_{jk}}{\partial x_1} = \psi'_{jk_1}(x_1)\phi_{jk_2}(x_2)$ and $\frac{\partial\theta^1_{jk}}{\partial x_2} = \psi_{jk_1}(x_1)\phi'_{jk_2}(x_2)$. Since the two-dimensional wavelets are separable in the x_1 and x_2 directions, so are the 2-D inner products. In other words,

$\left\langle I, \frac{\partial \theta^1_{jk}}{\partial x_1} \right\rangle = \left\langle I(x_1), \psi^0_{jk_1}(x_1) \right\rangle \left\langle I(x_2), \phi_{jk_2}(x_2) \right\rangle$. The …rst factor in this tensor product, $\left\langle I(x_1), \psi^0_{jk_1}(x_1) \right\rangle$, is de…ned so that we are taking grayscale intensity values pixel row by pixel row (i.e. $I(x_1) = I(x_1, x_2)$ where x_2 is …xed) to generate a 1-D inner product. In the second factor, $\left\langle I(x_2), \phi_{jk_2}(x_2) \right\rangle$, intensity values are being taken pixel column by pixel column (i.e. $I(x_2) = I(x_1, x_2)$ where x_1 is …xed). Likewise, we may write $\left\langle I, \frac{\partial \theta^1_{jk}}{\partial x_2} \right\rangle$ as a tensor product of one-dimensional inner products, $\left\langle I, \frac{\partial \theta^1_{jk}}{\partial x_2} \right\rangle = \left\langle I(x_1), \psi_{jk_1}(x_1) \right\rangle \left\langle I(x_2), \phi^0_{jk_2}(x_2) \right\rangle$. Now, we see that we may apply the previous analysis for computing the 1-D inner products with either ϕ, ψ or their derivatives at various levels of resolution j using the cascade algorithm. This provides all of the necessary components for the construction of S_{jk} [22], [3], [7], [28].

As the …nal implementation of Bernard's algorithm currently remains an un…nished product, we turn to the completely di¤erent approach to solving for optic ‡ow …rst proposed by Horn and Schunck [14].

7 Calculus of Variations Approach

7.1 An Additional Constraint Equation

Optic ‡ow is no di¤erent from any other question appearing in applied mathematics in that vastly di¤erent methods have been proposed to …nd a solution. While Bernard relied on a multiscale approach to treat the problem as a well-posed matrix equation [3], [4], [5], [6], [7], Horn and Schunck instead turn to the addition of an additional constraint upon the motion of grayscale intensity [14], [3].

The approach is founded upon the premise that sharp edges occurring in an image sequence cause competition between a large number of vectors representing the optic ‡ow. In other words, if two distinct objects which neighbor each other are moving with di¤erent velocities, it becomes very di¢cult to determine the motion of grayscale intensity along a sharp boundary between them [14]. This concept is illustrated in Figure 13.

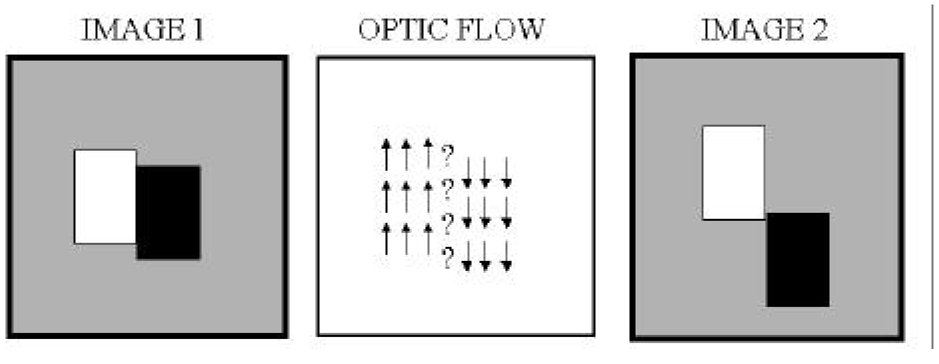

Figure 13: Sharp Edges Cause Competition

Here we have two images from a sequence in which a white rectangle borders a black rectangle with a gray background. The white rectangle moves up while the black one moves down. Describing the optic flow poses no problem except at the sharp edge which defines the boundary between the two objects. Along this edge, it becomes unclear whether the vector field should indicate an upward or a downward motion of grayscale intensity.

According to Horn and Schunck, this difficulty may be overcome by imposing a smoothness constraint upon the optic flow field. In essence, the constraint carries out a form of averaging to allow for the finding of one single "averaged" vector to represent the optic flow. The expense of such a technique is the loss of sharpness [14].

This smoothness constraint can be expressed as the minimization of the square of the magnitude of the gradient of the optic flow velocity. To ease the burden of notation for the following discussion, we adopt x and y as the Cartesian coordinates rather than x_1 and x_2. Furthermore, we refer to the optic flow as having components u and v in $\mathbf{v} = (u, v)$ whereas we had previously used v_1 and v_2. If we define the gradients $\nabla u = (u_x, u_y) = \left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}\right)$ and $\nabla v = (v_x, v_y) = \left(\frac{\partial v}{\partial x}, \frac{\partial v}{\partial y}\right)$, then the measure of the departure from smoothness in the velocity flow may be expressed as

$$\varepsilon_S^2 = \|\nabla u\|^2 + \|\nabla v\|^2 = \left(\sqrt{(u_x)^2 + (u_y)^2}\right)^2 + \left(\sqrt{(v_x)^2 + (v_y)^2}\right)^2$$

$$= u_x^2 + u_y^2 + v_x^2 + v_y^2.$$

Furthermore, we incorporate the intensity constraint equation (i.e. the governing equation for optic flow) and represent the error due to the grayscale intensity changing over local space and time as

$$\varepsilon_I = I_x u + I_y v + I_t,$$

where $I_x = \frac{\partial I}{\partial x}$, $I_y = \frac{\partial I}{\partial y}$, and $I_t = \frac{\partial I}{\partial t}$.

The goal then becomes to minimize the total error

$$\varepsilon^2 = \int\int \left(\varepsilon_I^2 + \delta \varepsilon_S^2\right) dxdy. \qquad (31)$$

This minimization of Equation (31) is achieved by finding the appropriate values for the optic flow velocity $\mathbf{v} = (u,v)$. Thus we are presented with a scheme for satisfying the constraint equations by solving for the vector field we seek. We go about such a task using a method called Calculus of Variations [14], [18], [13, Ch. 2].

7.2 Calculus of Variations

One of the key concepts in Calculus of Variations is what is known as a functional. Consider a set γ of functions which all satisfy certain conditions. A functional can be thought of as no more than a quantity which assumes a specific value corresponding to each function in γ. In essence, a functional is a function of functions. For example, the definite integral $I = \int_a^b f(x)dx$ is a functional since its value is determined by the function f [13, pg. 131].

The most fundamental problem in Calculus of Variations is to find the necessary conditions on u so that it minimizes the functional

$$I[u] = \iint_\Omega F\left(u, u_x, u_y\right) dxdy, \tag{32}$$

where F can be differentiated at least twice in both x and y. In order to find the necessary conditions on u, we consider the functional $J[\epsilon]$ defined as $I[u + \epsilon\eta]$ where u is the minimizer and η is any smooth function in the region Ω which vanishes on the boundary of Ω, denoted by $\partial\Omega$. In order for u to minimize Equation (32), $\epsilon = 0$ must minimize J. Therefore, we differentiate J with respect to ϵ and set it equal to 0 when $\epsilon = 0$. However,

$$J[\epsilon] = \iint_\Omega F\left(u + \epsilon\eta, u_x + \epsilon\eta_x, u_y + \epsilon\eta_y\right) dxdy, \tag{33}$$

which yields

$$J'[0] = \iint_\Omega \left(\eta F_u + \eta_x F_{u_x} + \eta_y F_{u_y}\right) dxdy = 0, \tag{34}$$

where $F_u = \dfrac{\partial F}{\partial u}$, $F_{u_x} = \dfrac{\partial F}{\partial u_x}$, and $F_{u_y} = \dfrac{\partial F}{\partial u_y}$.
We now note that

$$\eta_x F_{u_x} = -\eta(F_{u_x})_x + (\eta F_{u_x})_x, \quad \eta_x F_{u_x} = -\eta(F_{u_y})_y + (\eta F_{u_y})_y.$$

If we apply the definition of divergence, $div\langle m, n\rangle = \dfrac{\partial m}{\partial x} + \dfrac{\partial n}{\partial y}$, we see that

$$J'[0] = \iint_\Omega \eta\left(F_u - (F_{u_x})_x + (F_{u_y})_y\right) dxdy + \iint_\Omega div\left(\eta\left\langle F_{u_x}, F_{u_y}\right\rangle\right) dxdy = 0. \tag{35}$$

Next we apply the Divergence Theorem which states: $\iint_\Omega div(\mathbf{W})dxdy = \int_{\partial\Omega} (\mathbf{W}\cdot\mathbf{n})\, dt$ where $\mathbf{n}$ is the outward normal vector to $\partial\Omega$. This allows us to replace the second integral in Equation (35) with $\int_{\partial\Omega}\left(\eta\left\langle F_{u_x}, F_{u_y}\right\rangle\cdot\mathbf{n}\right) dt$. Since η was defined to satisfy zero boundary conditions on $\partial\Omega$, this integral vanishes. Therefore, we may write

$$J'[0] = \iint_\Omega \eta \left(F_u - (F_{u_x})_x + (F_{u_y})_y \right) dxdy = 0, \tag{36}$$

which applies for any smooth function η in Ω. In order for this to be the case, the integrand must itself be equal to zero. This provides the necessary condition on F and u for there to be a minimizer:

$$F_u - (F_{u_x})_x - (F_{u_y})_y = 0. \tag{37}$$

Equation (37) is referred to as the Euler-Lagrange equation for Equation (32) [18], [13, Ch. 2].

Let us now examine an example problem, namely Laplace's Equation. Consider the Dirichlet integral

$$J[u] = \iint_\Omega \left(u_x^2 + u_y^2 \right) dxdy. \tag{38}$$

Here we see that $F(u, u_x, u_y) = u_x^2 + u_y^2$. We now apply Equation (37),

$$F_u - (F_{u_x})_x - (F_{u_y})_y = 0$$

$$0 - (2u_x)_x - (2u_y)_y = 0$$

$$-2(u_{xx} + u_{yy}) = 0$$

So that we obtain

$$\nabla^2 u = 0, \tag{39}$$

where $\nabla^2 u = u_{xx} + u_{yy} = \dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}$ is the Laplacian of u [18], [13, Ch. 2].

We now seek to show that the same argument may be applied to a functional J which operates on a vector $\mathbf{u} = (u, v)$. In this case, we seek to minimize

$$J[u, v] = \iint_\Omega F(u, v, u_x, u_y, v_x, v_y) dxdy. \tag{40}$$

Applying the principle of the Euler-Lagrange equation to two dimensions we see that

$$F_u - (F_{u_x})_x - (F_{u_y})_y = 0, \qquad F_v - (F_{v_x})_x - (F_{v_y})_y = 0. \tag{41}$$

We now have the basic tools needed to apply Calculus of Variations to the two-dimensional context of optic flow stated as a minimization problem in Equation (31) [18], [13, Ch. 2].

7.3 Application of Calculus of Variations to Optic Flow

Recalling our de…nitions of ε_I and ε_S^2, we rewrite Equation (31) as a functional with arguments u and v representing the components of the optic ‡ow, **v** = (u, v).

$$J[u,v] = \iint_{\Omega} (I_t + I_x u + I_y v)^2 + \delta(u_x^2 + u_y^2 + v_x^2 + v_y^2)dxdy. \tag{42}$$

Since the term $\delta(u_x^2 + u_y^2 + v_x^2 + v_y^2)$ represents the departure from smoothness in the velocity ‡ow, it may be viewed as making the problem less "rigid" mathematically in that it allows for a certain amount (i.e. an amount proportional to the regularization parameter δ) of elastic stretching in the …eld which de…nes optic ‡ow [1].

We now apply Equation (41) and the result obtained from Laplace's Equation,

$$F_u - (F_{u_x})_x + (F_{u_y})_y = 0, \qquad F_v - (F_{v_x})_x + (F_{v_y})_y = 0$$

$$2I_x(I_t + I_x u + I_y v) - \delta(2\nabla^2 u) = 0, \quad 2I_y(I_t + I_x u + I_y v) - \delta(2\nabla^2 v) = 0$$

which yields

$$I_x(I_t + I_x u + I_y v) = \delta\nabla^2 u, \quad I_y(I_t + I_x u + I_y v) = \delta\nabla^2 v. \tag{43}$$

These are the same equations which Horn and Schunck attempt to solve for the optic ‡ow **v** = (u, v) [18], [14]. Notice that there are two equations with two unknowns, making the problem well-posed.

At this point, our method diverges from that of Horn and Schunck. We seek to apply the Galerkin method in order to …nd a numerical solution to the system of partial di¤erential equations in (43) whereas an approach based on …nite di¤erences is proposed in [14].

7.4 Insertion of the Galerkin Method

Our departure from the …nite di¤erence scheme presented by Horn and Schunck bears no relation to a lack of trust in their method. On the contrary, the theoretical development behind …nite di¤erences shows strong promise for application to such a problem as optic ‡ow [14], [10], [25], [15].

Rather, we choose the Galerkin method based on the fact that it seeks solutions of partial di¤erential equations in terms of a countable basis ϕ_i [18]. Extensive work has been done to prove the e¤ectiveness of the Galerkin or "spectral" method in solving PDEs using traditional bases such as Fourier (sines and cosines) or polynomials (Chebyshev, Legendre, or Hermite) [11], [21], [12], [25], [9], [18]. Thus the appeal of applying the Galerkin method to optic ‡ow, aside from the fact that it has never been done before, is that it lends itself to the employment of a wavelet basis. Although the strategy for making use of wavelets in this context differs greatly from Bernard's approach, there exists the possibility that they will open up a new door in the overall quest to solve the problem of optic flow. For now, we describe the method in terms of a generic countable basis ϕ_i.

Going back to Equation (36), we choose a special class of functions η, namely ϕ_i, with the end goal being to seek solutions of the form

$$u(x,y,t) = \sum_{j=1}\sum_{i=1} a_{i,\,j}(t)\phi_i(x)\phi_j(y), \quad v(x,y,t) = \sum_{j=1}\sum_{i=1} b_{i,\,j}(t)\phi_i(x)\phi_j(y). \tag{44}$$

Here, we see that the components of optic flow can be written as an exact sum of time dependent coefficients multiplying a two-dimensional spatially dependent basis. In practice, we are only able to compute a finite number of terms in Equation (44). If we use enough terms, then the equations in (44) can be approximated

$$u(x,y,t) \approx u_N(x,y,t) = \sum_{j=1}\sum_{i=1} a_{i,\,j}(t)\phi_i(x)\phi_j(y), \quad v(x,y,t) \approx v_N(x,y,t) = \sum_{j=1}\sum_{i=1} b_{i,\,j}(t)\phi_i(x)\phi_j(y). \tag{45}$$

Now, applying Equations (36) and (43), we see

$$\iint_\Omega \phi_i\phi_j \left[F_u - (F_{u_x})_x - (F_{u_y})_y\right]\Big|_{u=u_N,\ v=v_N} dxdy = 0, \quad i = 1,2,...,N \ \ and \ \ j = 1,2,...,N$$

$$\boxed{\iint_\Omega \phi_i\phi_j \left[I_x(I_t + I_xu + I_yv) - \delta\nabla^2 u\right]\Big|_{u=u_N,\ v=v_N} dxdy = 0, \quad i = 1,2,...,N \ and \ j = 1,2,...,N} \tag{46}$$

and

$$\iint_\Omega \phi_i\phi_j \left[F_v - (F_{v_x})_x - (F_{v_y})_y\right]\Big|_{u=u_N,\ v=v_N} dxdy = 0, \quad i = 1,2,...,N \ \ and \ \ j = 1,2,...,N$$

$$\boxed{\iint_\Omega \phi_i\phi_j \left[I_y(I_t + I_xu + I_yv) - \delta\nabla^2 v\right]\Big|_{u=u_N,\ v=v_N} dxdy = 0, \quad i = 1,2,...,N \ and \ j = 1,2,...,N.} \tag{47}$$

Equations (46) and (47) represent $2N^2$ equations with $2N^2$ unknowns. Thus, we have utilized the smoothness constraint equation, calculus of variations, and the Galerkin method to create a solvable, well-posed problem [18], [14]. The difficulty now lies in finding a feasible scheme for implementation in Mathematica®, our scientific computing program of choice for this method.

7.5 Issues Of Implementation

In both Equations (46) and (47), we observe the presence of the term I_t, the time derivative of the grayscale intensity function. The definition of this derivative states that

$$I_t = \frac{\partial I(x,y,t)}{\partial t} = \lim_{h \to 0} \frac{I(x,y,t+h) - I(x,y,t)}{h}.$$

A problem arises as one tries to implement the Galerkin approach to solve for optic flow in that the value of h cannot be taken to the limit of 0. The time step is fixed by the interval between discrete frames in the image sequence. Thus we must choose an approximation to I_t. For our purposes, we have very simply chosen $h = 1$ so that $I_t = I(x,y,t+1) - I(x,y,t)$. Although this is different from the approximation given in [14], no evidence has presented itself so far to show that our approach will specifically cause instability.

Another issue presents itself with relation to the spatial derivatives of the intensity function, I_x and I_y. In order to take these derivatives directly, we must have a function I which is continuous in the x and y directions. This is accomplished by interpolating the discrete grid of pixel intensity values obtained by importing a grayscale image into Mathematica®. The command which performs this task is called ListInterpolation.

The other spatial derivatives being taken, namely the Laplacians $\nabla^2 u$ and $\nabla^2 v$, do not pose such an obstacle because the components of the velocity can be represented in terms of differentiable basis functions, $u(x,y,t) = \sum_{j=1} \sum_{i=1} a_{i,j}(t)\phi_i(x)\phi_j(y)$, $v(x,y,t) = \sum_{j=1} \sum_{i=1} b_{i,j}(t)\phi_i(x)\phi_j(y)$.

Differentiability of the cosine function is one reason for our choice of the Fourier basis to serve as ϕ. The other reasoning comes from the fact that there already exists a solid foundation for use of this basis in the Galerkin method [11], [21], [12], [25], [9], [18]. There exists a definite problem, however, in applying periodic bases of infinite support, such as cosines, to functions which are of finite domain and are most likely non-periodic, such as the optic flow generated by a grayscale image sequence. Traditionally, to attain a good quality approximation of such a function using the basis $\{\phi_n(x) = \cos(n\pi x)\}_n$, a very large number of modes are required. This means that the "cut off" number N will have to be very large for $u \approx u_N$ and $v \approx v_N$. Thus our computation of the optic flow vectors which requires Mathematica® to solve $2N^2$ integral equations for $2N^2$ unknowns will take an extremely long amount of time and may even exhaust the memory for a standard computer [12], [26].

As was mentioned earlier, the Galerkin method can be executed using a variety of different choices for basis functions ϕ. The difficulty of approximating the optic flow, as described in the preceding paragraph, would be be better approached using a basis whose functions were indexed over both space and scale, such as wavelets [26], [16]. The only problem with using such a basis is that the approximations to the optic flow then become $u_N(x,y,t) = \sum_{j=1} \sum_{k_1=1} \sum_{k_2=1} a_{j,k_1,k_2}(t)\phi_{j,k_1}(x)\phi_{j,k_2}(y)$ and $v_N(x,y,t) =$

$\sum_{j=1}\sum_{k_1=1}\sum_{k_2=1} b_{j,\,k_1,\,k_2}(t)\phi_{j,\,k_1}(x)\phi_{j,\,k_2}(y)$ which when substituted into Equations (46) and (47) result in the solving of $2N^3$ equations for $2N^3$ unknowns. Consequently, the advantage of requiring less terms for a good approximation may not be gained after all.

As is the case at the completion of every research project, a number of issues still remain unresolved. Therefore, let us now turn to the results obtained using the protocol for implementation which is already in place.

8 Results

8.1 Wavelet Based Multiscale Approach

Results have not yet been attained using this approach. The remaining unresolved issues of implementation are extremely close to being overcome.

8.2 Calculus of Variations Approach

The first attempt at finding an optic flow vector field was conducted with a very simplified approximation to motion occurring between two images. Since a grayscale image is treated as a surface by a computer, we simply used known functions to create surfaces that would represent two pseudo-images. Our first image in the sequence was represented by the function $e1(x, y) = \cos(2\pi x)\sin(3\pi y)$ over the domain $[0, 1] \times [0, 1]$. In order to simulate motion we add to $e1$ a bump, and so create the second image represented by the function $e2(x,y) = e1(x, y) + g_{a,b}(x, y)$, where

$$g_{a,b}(x, y) = \begin{cases} \cos(4\pi \cdot r_{a,b}(x, y)) & if \quad r_{a,b}(x, y) = \sqrt{(x - a)^2 + (y - b)^2} < \frac{1}{8} \\ 0 & otherwise \end{cases}$$

and we set $a = 0.45$ and $b = 0.53$.

Figure 14 illustrates the first and second pseudo-images and then the bump which simulates the motion of grayscale intensity between the two.

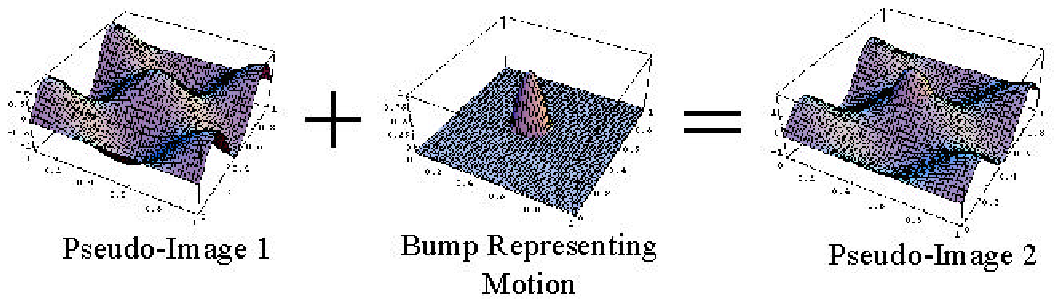

Figure 14: Simple Case Pseudo-Images and Bump

The corresponding optic ‡ow vector …elds were solved for with the Mathematica® notebook $HS1.nb$ using both 3 and 11 modes for the Fourier cosine basis. Figure 15 displays the results.

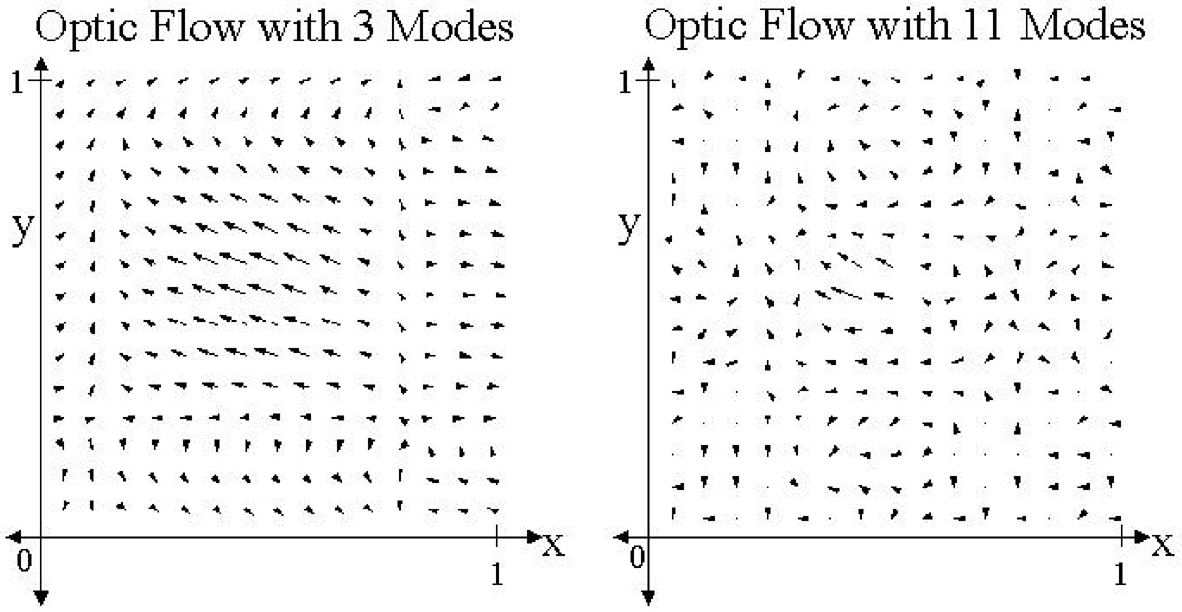

Figure 15: Optic Flow Between Simple Case Pseudo-Images

For both cases, there are non-trivial vectors occurring only in the region surrounding x = .45 and y = .53. The rest of the arrows are represented by Mathematica® as having no tails and so they can be assumed to be of value zero. This result is as it should be since the motion, simulated by the bump, only occurred in a region centered at $(.45, .53)$. Looking closer at the vector …elds, we see that the motion appears to be taking place over a larger area when we used 3 modes to solve for it as opposed to using 11. If we glance back to Figure 14, we see that the relative size of the bump in the domain $[0, 1]$ £ $[0, 1]$ more closely resembles the size of the region experiencing motion in the optic ‡ow …eld with a solution based on 11 modes. This provides evidence to support our conjecture that the approximation to the optic ‡ow is more accurate if we use more basis functions. The only problem is that the solution took signi…cantly more time …nd when Mathematica® had to solve the $2(11)^2$ = 242 equations instead of the $2(3)^2$ = 18 equations (approximately 30 minutes as opposed to 2 minutes).

We see that the basic premise of optic ‡ow has been put into place for a simpli…ed approximation to an image sequence. The next step is to …nd solutions to the optic ‡ow between two grayscale images in an actual sequence. This was accomplished using a slight variation from our method described earlier.

Previously, we used the command ListInterpolation to create functions which represented the images so that we could take the partial derivatives I_x and I_y. The presence of these interpolated functions is one of the leading factors which cause Mathematica® to take a long time to solve the integral equations in (46) and (47). Therefore, we attempt to circumvent this problem by using a Fourier cosine series to approximate the images according to

$$I(x,y) \approx I_N(x,y) = \sum_{n=1}\sum_{n=1} c_{m,n}\phi_m(x)\phi_n(y),$$

where the coe¢cients are computed by $c_{m,n} = \frac{\langle I(x,y), \phi_m(x)\phi_n(y)\rangle}{\langle \phi_m(x)\phi_n(y), \phi_m(x)\phi_n(y)\rangle}$.
Thus, despite all of the partial derivatives and basis functions contained in (46) and (47), the integrands of these equations become nothing more than a multiplication between sines and cosines. Solving a system such as this is much less computationally intensive since the integrals can be computed exactly without the need to consider the e¤ects of interpolation. The drawback to this technique is that accuracy is lost in performing a cosine approximation of an image. Sharp transitions in grayscale intensity are di¢cult to capture using functions which are smooth and of in…nite support. The bene…t, however, of this approach is that we are able to use more frequency modes in the approximation while maintaining the time required to …nd a solution to the optic ‡ow.

Figure 16 illustrates the fact that the general trend of motion can be captured with this method using 16 modes of frequency. The optic ‡ow is found using the Mathematica® notebook HSGalerkin.nb.

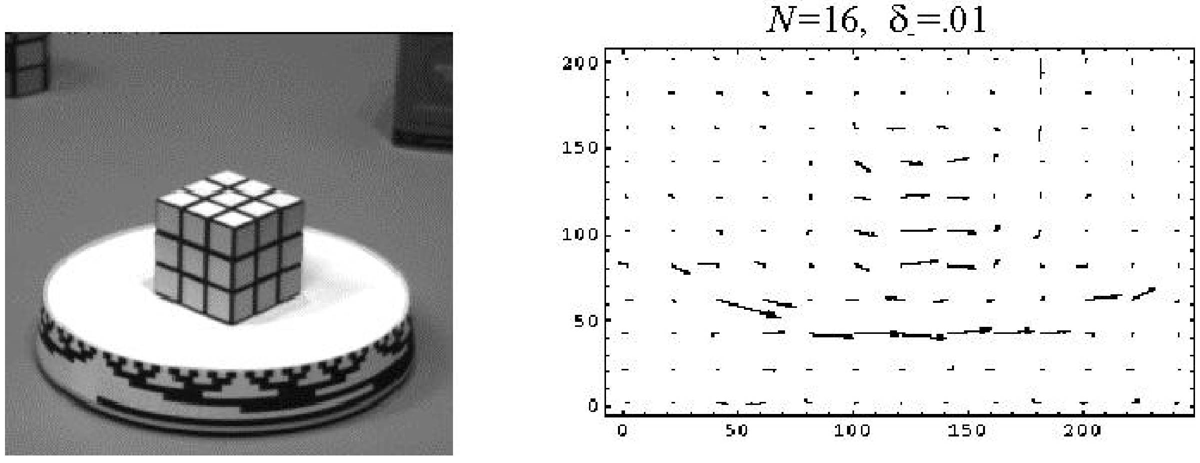

Figure 16: Optic Flow Using the Galerkin Method

9 Conclusions

We have explored some of the fundamental theory which lies behind two di¤erent methods of solving for optic ‡ow. Additionally, a solid path has been laid down for the process of implementing each technique. Many riddles and obstacles that existed before have now been resolved or at least have a starting point from which to approach them.

It remains inconclusive as to which is a better method, the wavelet based multiscale approach proposed by Bernard or the calculus of variations scheme developed by Horn and Schunck. On one hand, Bernard's technique appears to be more robust in its ability to address the issues of time aliasing and aperture. The multiresolutional nature of the wavelet basis seems to make it the perfect tool for dealing with such a problem. A weakness arises in Bernard's argument, however. He makes the assertion that by carrying out the multiresolutional decomposition all the way to the coarsest level, the solution to the system of matrix equations will converge to the actual optic ‡ow. The problem stems from his key assumption that the optic ‡ow is constant over the support of the wavelets [3], [4], [5], [6], [7]. At the coarser levels of resolution, the wavelets have support which span a large portion of the image. That the optic ‡ow would be uniform throughout so large an area is not a realistic statement. Yet without this assumption in place, the matrix equations cannot be constructed. Therefore, from a sense of practicality, we must end the multiresolutional decomposition at the level where the optic ‡ow begins to violate this assumption. The question as to whether or not the solution of this abridged system will still converge to the true optic ‡ow is left unanswered by Bernard. We see that here he has left an enormous distance between the theory and the practical implementation of his method.

On the other hand, the approach of Horn and Schunck appears to make more sense from a standpoint of physics. The key assumption made by Bernard about the constancy of optic ‡ow over various local scales is replaced with the much more plausible notion that the optic ‡ow velocity makes up a smooth …eld. Consider Figure 13 again where two neighboring objects are undergoing di¤erent motion. The smoothness constraint of Horn and Schunck forces the optic ‡ow to take on an "averaged" value at the boundary between the two objects, thus losing the quality of sharpness [14]. Such a loss of information can be considered much less signi…cant, however, than that experienced by having to classify the motion as one constant value for the area taken up by the entirety of both objects as would happen under Bernard's method. Problems are encountered with the calculus of variations algorithm, however, when the image sequence is complicated by the appearance of numerous small objects travelling with distinct velocities either in close proximity or overlapping each other. In a case such as this, the loss of sharpness experienced at the borders of the objects is much more substantial since the objects may not comprise su¢ cient pixel area to preserve their true motion from the averaging requirement imposed by the smoothness constraint. The problem with this fact is that almost any video sequence could realistically contain this kind of complexity. Thus, we see that this method may only have practicality for simple image sequences.

Of course, the development of each method has not reached a …nal end. Numerous opportunities exist to adjust each model to better suit the problem of optic ‡ow. For instance, Bernard's research has begun to explore the concept of "image warping". This entails estimating the dynamic of motion as a uniform translation across the image (i.e. all objects travel with the same velocity) and then solving for the deviation from this warped image at various scales. The assertion is that if enough scales are used in the multiresolution, the sum of the large motion estimates with the residual deviations will be su¢ciently close to the true motion [7]. As was mentioned earlier, the use of the Galerkin method to solve the calculus of variations problem developed by Horn and Schunck might be improved if a more high-powered basis was used such as one incorporating wavelets. It is a distinct possibility that there exists some trick which relies on the orthonormality of wavelets to make the solving of the integral equations replaceable by a much less computationally intensive task [28].

In addition, some of the fundamental issues relating to optic ‡ow are left unresolved. For instance, let us say that one was attempting to track the real motion of objects by …lming them and then solving for the corresponding optic ‡ow. How does one describe the di¤erence between the actual and the projected motion? The method for overcoming this obstacle, as mentioned earlier, is presented in [23] as an issue of stochastic error. Simoncelli states that it is possible to use an uncertainty model to express the departure of the optic ‡ow …eld from the real world scene of motion. It may well prove that this approach does a reasonably good job of addressing the problem. A highly accurate and adaptive stochastic model would have to be developed …rst, however, to apply this idea of motion sensing based on optic ‡ow to a context relevant to the military such as target tracking.

Another signi…cant yet unresolved issue is that the basic physics behind the optic ‡ow model might not hold. It is possible that the theory behind the governing equation (i.e. that grayscale intensity is conserved) is not valid. Bernard makes an attempt to address this concern by proposing a Lambertian surface aspect model to take into account illumination changes [3]. Although this idea makes intuitive sense as an improvement on the way to solve for optic ‡ow, it is not directly stated in his publications that better results were attained through the use of it.

Overall, it may be stated that optic ‡ow has a long way to go before it may stand alongside some of the problems in applied mathematics which are regarded as high powered and crucial for solving. The approaches taken to resolve the issues have only been explored to an extent where optic ‡ow may be applied in a limited sense. Questions of theory are not met with answers of implementation across a wide range of issues. This project has closed some of the gaps in the pursuit of providing more potential for application of optic ‡ow. However, much work still remains for future studies to build higher upon the foundations. It will be interesting to observe how far research is able to advance the concept and what it will be applied to.

It is the hope of the author that some day enough will be known to harness optic ‡ow for use in the domain of acoustic imagery supplied by sonar systems.

.

10 Bibliography

[1] Battiti, Roberto, Amaldi, Edoardo & Kock, Christof. Computing Optical Flow Across Multiple Scales: An Adaptive Coarse-to-Fine Strategy. International Journal of Computer Vision, 6:2: 133-145, 1991.

[2] Beauchemin, S.S. & Barron, J.L. The Computation of Optical Flow. ACM Computing Surveys, Vol. 27, No. 3: 433-467, 1995.

[3] Bernard, Christophe. Wavelets and ill-posed problems: optic ‡ow estimation and scattered data interpolation. Ph.D. Thesis, École Polytechnique, November 24th, 1999.

[4] Bernard, Christophe. Fast Optic Flow Computation.
http://www.cmap.polytechnique.fr/~bernard/OpticFlow/index.html?? , October 6th, 1998.

[5] Bernard, Christophe. Discrete Wavelet Analysis for Fast Optic Flow Computation. Rapport Interne du Centre de Mathématiques Appliquées RI415, École Polytechnique, February, 1999.
http://www.cmap.polytechnique.fr/~bernard/Publications/RI415.pdf??

[6] Bernard, Christophe. Fast Optic Flow Computation with Discrete Wavelets. Rapport Interne du Centre de Mathématiques Appliquées RI365, École Polytechnique, May, 1997.
http://www.cmap.polytechnique.fr/~bernard/Publications/bernard_365.mai.pdf??

[7] Bernard, Christophe. Discrete Wavelet Analysis for Fast Optic Flow Computation. Applied and Computational Harmonic Analysis, 11: 32-63, 2001.

[8] Daubechies, Ingrid. Ten Lectures on Wavelets. Philadelphia, Society for Industrial and Applied Mathematics, 1992.

[9] Fletcher, C.A.J. Computational Galerkin Methods. New York: Springer Series in Computational Physics, Springer-Verlag, 1984.

[10] Gollub, Gene & Ortega, James. Scienti…c Computing and Di¤erential Equations: An Introduction to Numerical Methods. San Diego: Academic Press, 1992.

[11] Gottlieb, David & Orszag, Steven. Numerical Analysis of Spectral Methods: Theory and Applications. Philadelphia: CBMS-NSF Regional Conference Series in Applied Mathematics, Society for Industrial and Applied Mathematics, 1997.

[12] Heineike, Benjamin. Modeling Morphogenesis with Reaction-Di¤usion Equations using Galerkin Spectral Methods. Annapolis, MD: Trident Scholar Report, United States Naval Academy, 2002.

[13] Hildebrand, Francis. Methods of Applied Mathematics, 2nd Ed. Englewood Cli¤s, NJ: Prentice-Hall, 1965.

[14] Horn, Berthold & Schunck, Brian. Determining Optic Flow. Arti…cial Intelligence, 17: 185-203, 1981.

[15] Ingle, Vinay & Proakis, John. Digital Signal Processing using MATLAB®. New York: Brooks/Cole, 2000.

[16] Kaiser, Gerald. A Friendly Guide to Wavelets. Boston: Birkhäuser, 1994.

[17] Malek-Madani, Reza. Advanced Engineering Mathematics: With Mathematica and Matlab, Vol. 2. Addison Wesley Longman, 1998.

[18] Malek-Madani, Reza. Introductory Notes on Calculus of Variations and Optic Flow. March, 2003.

[19] Mallat, Stéphane. A Wavelet Tour of Signal Processing, 2nd Ed. Boston: Academic Press, 1999.

[20] Nievergelt, Yves. Wavelets Made Easy. Boston: Birkhäuser, 2001.

[21] Orszag, Steven. Spectral Methods for Problems in Complex Geometries. Journal of Computational Physics, 37: 70-92, 1980.

[22] Pierce, John. Introductory Notes on the Cascade Algorithm With Derivative Scaling Function and Wavelet Bases. March, 2003.

[23] Simoncelli, Eero. Bayesian Multi-Scale Di¤erential Optical Flow. Handbook of Computer Vision and Applications, Vol. 2, Ch. 14: 397-422, 1999.

[24] Strang, Gilbert & Nguyen, Truong. Wavelets and Filter Banks. Wellesley, MA: Wellesley-Cambridge Press, 1996.

[25] Trefethen, Lloyd. Spectral Methods in MATLAB®. Philadelphia: Society for Industrial and Applied Mathematics, 2000.

[26] Walker, James. A Primer On Wavelets and Their Scienti…c Applications. New York: Chapman & Hall/CRC, 1999.

[27]Weber, Joseph & Malik, Jitendra. Robust Computation of Optical Flow in a Multi-Scale Di¤erential Framework. International Journal of Computer Vision, 14: 67-81, 1995.

[28] Wolfram Research, Inc. Wavelet Explorer Tutorial. Wolfram Research, Inc.: 1997.

11 Appendix

MATLAB® m-files

Haar1.m

```
%Used to compute Level-1 Haar Transform

function [alevel1,dlevel1]=Haar1(f);

%Input f is the sample function to be decomposed

N=length(f);

%Index j is the scale or level

j=1;

%Creation of matrix W to store 1-level Haar wavelets

W=zeros(N/2,N);

for m=1:N/2

    W(m,m*2^j-(2^j-1):m*2^j)=[ones(1,2^(j-1))/(2^(j/2)),-1*ones(1,2^(j-1))/(2^(j/2))];

end

%Creation of matrix V to store 1-level Haar scaling functions

V=zeros(N/2,N);

for m=1:N/2

    V(m,m*2^j-(2^j-1):m*2^j)=ones(1,2^j)/(2^(j/2));

end

%Finding vectors alevel1=a1,a2,...,aN/2 and dlevel1=d1,d2,...,dN/2

for m=1:N/2

    a(m)=f*V(m,:)';

    d(m)=f*W(m,:)';

end

alevel1=a;

dlevel1=d;
```

Haar10.m

```
%Used to compute Level-10 Haar Transform & plot Averaged Signals
function A=Haar10(f);
%Input f is the sample function to be decomposed
N=length(f);
%Creation of matrix W to store Haar wavelets
W=zeros(N/2,N,10);
for j=1:10
    for m=1:N/(2^j)
         W(m,m*2^j-(2^j-1):m*2^j,j)=[ones(1,2^(j-1))/(2^(j/2)),-1*ones(1,2^(j-1))/(2^(j/2))];
    end
end
%Creation of matrix V to store Haar scaling functions
V=zeros(N/2,N,10);
for j=1:10
    for m=1:N/(2^j)
        V(m,m*2^j-(2^j-1):m*2^j,j)=ones(1,2^j)/(2^(j/2));
    end
end
%Finding vectors alevelj and dlevelj
for j=1:10
    for m=1:N/(2^j)
        a(m,j)=f*V(m,:,j)';
        d(m,j)=f*W(m,:,j)';
    end
end
%Finding the …rst 10 averaged signals A1 to A10
A=zeros(10,N);
for j=1:10
    temp=zeros(1,N);
    for m=1:N/(2^j)
        temp=temp+a(m,j)*V(m,:,j);
        end
        A(j,:)=temp;
end
%Temporary code
%atemp=a(…nd(a(:,2)~=0),2); dtemp=d(…nd(d(:,2)~=0),2);
%A=[atemp',dtemp',d(:,1)'];
%Temporary code
a10temp=a(1:2^0,10);
d10temp=d(1:2^0,10); d9temp=d(1:2^1,9); d8temp=d(1:2^2,8);
d7temp=d(1:2^3,7); d6temp=d(1:2^4,6); d5temp=d(1:2^5,5);
d4temp=d(1:2^6,4); d3temp=d(1:2^7,3); d2temp=d(1:2^8,2);
A=[a10temp',d10temp',d9temp',d8temp',d7temp',d6temp',d5temp',
          d4temp',d3temp',d2temp',d(:,1)'];
```

Daub4_10.m

```
%Used to compute Level-10 Daub4 Transform & plot Averaged and Detail Signals
function [A,D]=Daub4_10(f);
%Input f is the sample function to be decomposed
N=length(f);
%Creation of matrix V to store Daub4 scaling functions
V=zeros(N/2,N,10);
%Scaling Numbers alpha1, alpha2, alpha3, alpha4
alpha1=(1+sqrt(3))/(4*sqrt(2)); alpha2=(3+sqrt(3))/(4*sqrt(2));
alpha3=(3-sqrt(3))/(4*sqrt(2)); alpha4=(1-sqrt(3))/(4*sqrt(2));
%Create 1st level scaling functions, then from this generate the next levels in the MRA
V(1,1:4,1)=[alpha1,alpha2,alpha3,alpha4];
for m=2:(N/2)
        V(m,:,1)=[V(1,N-2*(m-1)+1:N,1),V(1,1:N-2*(m-1),1)];
end
%Create 2nd through 10th level scaling functions from 1st level scaling functions
for j=2:10
        for m=1:N/(2^j)-1
        V(m,:,j)=alpha1*V(2*m-1,:,j-1)+alpha2*V(2*m,:,j-1)+...
        alpha3*V(2*m+1,:,j-1)+alpha4*V(2*m+2,:,j-1);
    end
    %Account for wrap around [V_2m+1 & V_2m+2 will exceed N/(2^j)]
    m=N/(2^j);
    V(m,:,j)=alpha1*V(2*m-1,:,j-1)+alpha2*V(2*m,:,j-1)+...
    alpha3*V(2*m+1-N/(2^(j-1)),:,j-1)+alpha4*V(2*m+2-N/(2^(j-1)),:,j-1);
end
%Creation of matrix W to store Daub4 Wavelets
W=zeros(N/2,N,10);
%Scaling Numbers beta1, beta2, beta3, beta4
beta1=(1-sqrt(3))/(4*sqrt(2)); beta2=(sqrt(3)-3)/(4*sqrt(2));
beta3=(3+sqrt(3))/(4*sqrt(2)); beta4=(-1-sqrt(3))/(4*sqrt(2));
%Create 1st level Wavelets, then from this generate the next levels in the MRA
W(1,1:4,1)=[beta1,beta2,beta3,beta4];
for m=2:(N/2)
    W(m,:,1)=[W(1,N-2*(m-1)+1:N,1),W(1,1:N-2*(m-1),1)];
end
%Create 2nd through 10th level Wavelets from 1st level Wavelets
for j=2:10
    for m=1:N/(2^j)-1
        W(m,:,j)=beta1*V(2*m-1,:,j-1)+beta2*V(2*m,:,j-1)+...
        beta3*V(2*m+1,:,j-1)+beta4*V(2*m+2,:,j-1);
    end
    %Account for wrap around [W_2m+1 & W_2m+2 will exceed N/(2^j)]
    m=N/(2^j);
    W(m,:,j)=beta1*V(2*m-1,:,j-1)+beta2*V(2*m,:,j-1)+...
    beta3*V(2*m+1-N/(2^(j-1)),:,j-1)+beta4*V(2*m+2-N/(2^(j-1)),:,j-1);
end
%Finding vectors alevelj and dlevelj
for j=1:10
    for m=1:N/(2^j)
```

```
            a(m,j)=f*V(m,:,j)';
            d(m,j)=f*W(m,:,j)';
        end
end
%Finding the …rst 10 averaged signals A1 to A10
A=zeros(10,N);
for j=1:10
        temp=zeros(1,N);
        for m=1:N/(2^j)
            temp=temp+a(m,j)*V(m,:,j);
        end
        A(j,:)=temp;
end
%Finding the …rst 10 detail signals D1 to D10
D=zeros(10,N);
for j=1:10
        temp=zeros(1,N);
        for m=1:N/(2^j)
            temp=temp+d(m,j)*W(m,:,j);
        end
        D(j,:)=temp;
end
```

MakeDaub4_10.m

```
%Purpose is simply to make various Daub4 scaling functions and wavelets on their own
function [V,W]=MakeDaub4_10;
%Arbitrarily set N=1024 since many of the signals we have been using are this length
N=2^10;
%Creation of matrix V to store Daub4 scaling functions
V=zeros(N/2,N,10);
%Scaling Numbers alpha1, alpha2, alpha3, alpha4
alpha1=(1+sqrt(3))/(4*sqrt(2)); alpha2=(3+sqrt(3))/(4*sqrt(2));
```

```
alpha3=(3-sqrt(3))/(4*sqrt(2)); alpha4=(1-sqrt(3))/(4*sqrt(2));
%Create 1st level scaling functions, then from this generate the next levels in the MRA
V(1,1:4,1)=[alpha1,alpha2,alpha3,alpha4];
for m=2:(N/2)
    V(m,:,1)=[V(1,N-2*(m-1)+1:N,1),V(1,1:N-2*(m-1),1)];
end
%Create 2nd through 10th level scaling functions from 1st level scaling functions
for j=2:10
    for m=1:N/(2^j)-1
        V(m,:,j)=alpha1*V(2*m-1,:,j-1)+alpha2*V(2*m,:,j-1)+...
        alpha3*V(2*m+1,:,j-1)+alpha4*V(2*m+2,:,j-1);
    end
    %Account for wrap around [V_2m+1 & V_2m+2 will exceed N/(2^j)]
    m=N/(2^j);
    V(m,:,j)=alpha1*V(2*m-1,:,j-1)+alpha2*V(2*m,:,j-1)+...
    alpha3*V(2*m+1-N/(2^(j-1)),:,j-1)+alpha4*V(2*m+2-N/(2^(j-1)),:,j-1);
end
%Creation of matrix W to store Daub4 Wavelets
W=zeros(N/2,N,10);
%Scaling Numbers beta1, beta2, beta3, beta4
beta1=(1-sqrt(3))/(4*sqrt(2)); beta2=(sqrt(3)-3)/(4*sqrt(2));
beta3=(3+sqrt(3))/(4*sqrt(2)); beta4=(-1-sqrt(3))/(4*sqrt(2));
%Create 1st level Wavelets, then from this generate the next levels in the MRA
W(1,1:4,1)=[beta1,beta2,beta3,beta4];
for m=2:(N/2)
    W(m,:,1)=[W(1,N-2*(m-1)+1:N,1),W(1,1:N-2*(m-1),1)];
end
%Create 2nd through 10th level Wavelets from 1st level Wavelets
for j=2:10
    for m=1:N/(2^j)-1
        W(m,:,j)=beta1*V(2*m-1,:,j-1)+beta2*V(2*m,:,j-1)+...
        beta3*V(2*m+1,:,j-1)+beta4*V(2*m+2,:,j-1);
    end
    %Account for wrap around [W_2m+1 & W_2m+2 will exceed N/(2^j)]
    m=N/(2^j);
    W(m,:,j)=beta1*V(2*m-1,:,j-1)+beta2*V(2*m,:,j-1)+...
    beta3*V(2*m+1-N/(2^(j-1)),:,j-1)+beta4*V(2*m+2-N/(2^(j-1)),:,j-1);
end
```

Mathematica® notebooks

HS1.nb

```
%This notebook …nds the optic ‡ow for a very simple approximation to motion,
%    i.e. motion from one "image" to another is represented as a bump being
%    added to the surface describing the grayscale intensity of the …rst image.
<<Graphics`PlotField`;
% The parameter nn determines how many modes will be
%     used in the approximation, here nn=2.
nn=2;
% This creates the basis for approximating u and v
phi[i_,x_]=Cos[i Pi x];
u = Sum[a[i,j]phi[i,x]phi[j,y],{i,0,nn},{j,0,nn}];
v = Sum[b[i,j]phi[i,x]phi[j,y],{i,0,nn},{j,0,nn}];
% This creates the two "images" in the sequence, e1 and e2,
%      the second being the …rst with a bump added
%      to represent motion of grayscale intensity.
a1=0.45;b1=0.53;
r[x_,y_,a1_,b1_]=Sqrt[(x-a1)^2+(y-b1)^2];
e1=Sin[3 Pi y]Cos[2 Pi x];
e2=e1 + If[r[x,y]<1/8,Cos[4 Pi r[x,y,0.45.0.53]],0];
% op1 and op2 are the Euler-Lagrange equations where the parameter eps = δ.
op1=D[e1,x]^2 u + D[e1,x]D[e1,y] v - eps (D[u,{x,2}]+D[u,{y,2}]) + D[e1,x](e2-e1);
op2=D[e1,y]^2 v + D[e1,x]D[e1,y] u - eps (D[v,{x,2}]+D[v,{y,2}]) + D[e1,y](e2-e1);
```

```
% This sets up the integral equations to be solved by the Galerkin method
Do[term=Expand[op1 phi[i,x]phi[j,y]];equation1[i,j]=Map[Integrate[#, {x,0,1},{y,0,1}]&,term]==0;
      Print[Date[]];Print["i = ", i, ", j = ",j],{i,0,nn},{j,0,nn}];
Do[term=Expand[op2 phi[i,x]phi[j,y]];equation2[i,j]=Map[Integrate[#, {x,0,1},{y,0,1}]&,term]==0;
      Print[Date[]];Print["i = ", i, ", j = ",j],{i,0,nn},{j,0,nn}];
coe¤=Flatten[Table[{a[i,j],b[i,j]},{i,0,nn},{j,0,nn}]];
eqns=Flatten[Join[Table[equation1[i,j],{i,0,nn},{j,0,nn}],Table[equation2[i,j],{i,0,nn},{j,0,nn}]]];
% This solves the integral equations and plots the optic ‡ow vector …eld
eps=0.01;
sol=Solve[eqns,coe¤];
uu[x_,y_] = First[u/.sol];
vv[x_,y_] = First[v/.sol];
PlotVectorField[{uu[x,y],vv[x,y]},{x,0,1},{y,0,1}]
```

HSGalerkin.nb

```
% This notebook …nds the optic ‡ow for the Rubik’s Cube sequence
%     using the Galerkin method in the Calculus of Variations approach
<<Graphics‘PlotField‘
<<Graphics‘Arrow‘
g=Import["C:nnWindowsnnDesktopnnHornSchuncknnrubikseq.gif"];
% The images are …rst interpolated and then the partial derivatives
%     are taken and then multiplied to form the terms in the integral equations
e1=ListInterpolation[g[[1,1,1]],{{0,1},{0,1}}];
e3=ListInterpolation[g[[3,1,1]],{{0,1},{0,1}}];
et[x_,y_]=e3[x,y]-e1[x,y];
ex[x_,y_]=D[e1[x,y],x];
ey[x_,y_]=D[e1[x,y],y];
ex2[x_,y_]=ex[x,y]^2;
ey2[x_,y_]=ey[x,y]^2;
exey[x_,y_]=ex[x,y]ey[x,y];
exet[x_,y_]=ex[x,y]et[x,y];
eyet[x_,y_]=ey[x,y]et[x,y];
```

```
<<ex2.sav;
<<ey2.sav;
<<exet.sav;
<<eyet.sav;
<<exey.sav;
(*EX2=Table[ex2[x,y],{x,0,1,1/240},{y,0,1,1/256}];
EY2=Table[N[ey2[x,y]],{x,0,1,1/240},{y,0,1,1/256}];
EXEY=Table[N[exey[x,y]],{x,0,1,1/240},{y,0,1,1/256}];
EXET=Table[N[exet[x,y]],{x,0,1,1/240},{y,0,1,1/256}];
EYET=Table[N[eyet[x,y]],{x,0,1,1/240},{y,0,1,1/256}];*)
X=Table[x,{x,0,1,N[1/240]}];Y=Table[y,{y,0,1,N[1/256]}];
(*PHI[i_,j_]=Table[Cos[i Pi x]Cos[j Pi y],{x,0,1,N[1/240]},{y,0,1,N[1/256]}];*)
PHI[i_,j_]=Outer[Times,Cos[i Pi X],Cos[j Pi Y]];
% The cosine basis functions are set up to use 17 modes of frequency to approximate
%     the optic ‡ow vectors
nn=16;phi[i_,x_]=Cos[i Pi x];
u=Sum[a[i,j]phi[i,x]phi[j,y],{i,0,nn},{j,0,nn}];
v=Sum[b[i,j]phi[i,x]phi[j,y],{i,0,nn},{j,0,nn}];

% The cosine approximations for the terms in the integral equations are found
(*Print[Date[]];
Do[coe¤exet[k,l]=1/240 1/256 Plus @@ Flatten[EXET
PHI[k,l]],{k,0,nn},{l,0,nn}];
Print[Date[]];
Do[coe¤eyet[k,l]=1/240 1/256 Plus @@ Flatten[EYET
PHI[k,l]],{k,0,nn},{l,0,nn}];
Print[Date[]];
Do[coe¤exey[k,l]=1/240 1/256 Plus @@ Flatten[EXEY
PHI[k,l]],{k,0,nn},{l,0,nn}];
Print[Date[]];
Do[coe¤ex2[k,l]=1/240 1/256 Plus @@ Flatten[EX2
PHI[k,l]],{k,0,nn},{l,0,nn}];
```

```
Print[Date[]];
Do[coe¤ey2[k,l]=1/240 1/256 Plus @@ Flatten[EY2
PHI[k,l]],{k,0,nn},{l,0,nn}];
COEFFEXET=Table[coe¤exet[k,l],{k,0,nn},{l,0,nn}];
Save["COEFFEXET.sav",COEFFEXET];
COEFFEYET=Table[coe¤eyet[k,l],{k,0,nn},{l,0,nn}];
Save["COEFFEYET.sav",COEFFEYET];
COEFFEXEY=Table[coe¤exey[k,l],{k,0,nn},{l,0,nn}];
Save["COEFFEXEY.sav",COEFFEXEY];
COEFFEX2=Table[coe¤ex2[k,l],{k,0,nn},{l,0,nn}];
Save["COEFFEX2.sav",COEFFEX2];
COEFFEY2=Table[coe¤ey2[k,l],{k,0,nn},{l,0,nn}];
Save["COEFFEY2.sav",COEFFEY2];*)
<<COEFFEX2.sav;
<<COEFFEY2.sav;
<<COEFFEXEY.sav;
<<COEFFEXET.sav;
<<COEFFEYET.sav;

% The integral for a multiplication of 3 basis functions is carried out in advance
%     so that it does not have to be computed each time in the future
triple=Table[Integrate[phi[i,x]phi[j,x]phi[k,x],{x,0,1}],{i,0,nn},{j,0,nn},{k,0,nn}];

% The integral equations are set up
Do[equation1[k,l]=COEFFEXET[[k+1,l+1]]+Sum[COEFFEX2[[i+1,j+1]]a[m,n]triple[[i+
1,m+1,k+1]]triple[[j+1,n+1,l+1]],{i,0,nn},{j,0,nn},{m,0,nn},{n,0,nn}]+Sum[
COEFFEXEY[[i+1,j+1]]b[m,n]triple[[i+1,m+1,k+1]]triple[[j+1,n+1,l+1]],{i,0,nn},
{j,0,nn},{m,0,nn},{n,0,nn}] +
eps/4 ((k Pi)^2+(l Pi)^2) a[k,l]==0,{k,0,nn},{l,0,nn}];
Print[Date[],", equation1 is done"];
Do[equation2[k,l]=COEFFEYET[[k+1,l+1]]+Sum[COEFFEXEY[[i+1,j+1]]a[m,n]triple[[
i+1,m+1,k+1]]triple[[j+1,n+1,l+1]],{i,0,nn},{j,0,nn},{m,0,nn},{n,0,nn}]+Sum[
COEFFEY2[[i+1,j+1]]b[m,n]triple[[i+1,m+1,k+1]]triple[[j+1,n+1,l+1]],{i,0,nn},{
```

```
j,0,nn},{m,0,nn},{n,0,nn}] +
eps/4((k Pi)^2+(l Pi)^2) b[k,l]==0,{k,0,nn},{l,0,nn}];
Print[Date[]," , equation2 is done"];
coe¤=Flatten[Table[{a[i,j],b[i,j]},{i,0,nn},{j,0,nn}]];
eqns=Flatten[Join[Table[equation1[i,j],{i,0,nn},{j,0,nn}],Table[equation2[i,j],{i,0,nn},{j,0,nn}]]];
```

% The optic ‡ow vectors are solved for using δ = .01

```
eps=0.01;
sol=Solve[eqns,coe¤];
Print[Date[]," sol is done"];
uu[x_,y_]=First[u/.sol];
vv[x_,y_]=First[v/.sol];
discu=Table[uu[x,y],{x,0,1,1/240},{y,0,1,1/256}];
discv=Table[vv[x,y],{x,0,1,1/240},{y,0,1,1/256}];
```

% The optic ‡ow …eld is plotted

```
scale=Max[Table[Sqrt[discu[[i,j]]^2+discv[[i,j]]^2],{i,2,239},{j,2,255}]];
graph=Show[Table[Graphics[Arrow[{j,i},{j,i}+
100/scale{discv[[i,j]],discu[[i,j] ]},HeadScaling->Relative], Frame->True,
PlotLabel->StringJoin["Galerkin Method, eps = ", ToString[eps], ", n = ",ToString[nn]]],
{i,2,220,20},{j,2,256,20}]]
```